WHOI-98-13

Woods Hole Oceanographic Institution

Vorticity Measurements within the Bottom Boundary Layer in the Strait of Juan De Fuca

by

J.J. Fredericks, John H. Trowbridge, A.J. Williams, 3rd

July 1998

Technical Report

Funding was provided by the Office of Naval Research through
Grant No. N00014-94-I-0436.

Approved for public release; distribution unlimited.

WHOI-98-13

Vorticity Measurements within the Bottom Boundary Layer in the Strait of Juan De Fuca

by

J.J. Fredericks, John H. Trowbridge, A.J. Williams, 3rd

Woods Hole Oceanographic Institution
Woods Hole, Massachusetts 02543

July 1998

Technical Report

Funding was provided by the Office of Naval Research through
Grant No. N00014-94-I-0436.

Approved for public release; distribution unlimited.

Approved for Distribution:

Timothy Stanton, Chair
Department of Applied Ocean Physics and Engineering

Table of Contents

List of Tables

List of Figures

1. INTRODUCTION

A collaborative program to test the hypothesis that electromagnetic fluctuations at the sea floor are forced by turbulent vorticity fluctuations in the bottom boundary layer began in Fall, 1993. After testing of the instrumentation in the early part of the year, vorticity data were recorded, together with electromagnetic fluctuations, in the strong tidal flows in the Strait of Juan de Fuca through four seasons, Summer 1994 through Spring 1995. The vorticity measurements provide data for testing existing theoretical predictions of the spectrum of the vorticity fluctuations at wave numbers within the inertial subrange, and they can provide a means to characterize the vorticity spectrum at wave numbers below the inertial subrange.

The Woods Hole Oceanographic Institution (WHOI) component of the two-year program was to provide measurements of vorticity at two scales, 45 cm and 150 cm, together with measurements of horizontal velocities between 1 and 3.4 m above bottom. The Scripps Oceanographic Institution (SIO) component of the program was to simultaneously measure electromagnetic fluctuations near the WHOI vorticity meters. SIO also observed velocity from three nearby moorings between 3 m and 30 m above bottom. Conductivity-temperature-depth (CTD) casts were taken by SIO during deployment and several months later. A side-scan sonar survey of the region within a tidal excursion of the deployment site was conducted by WHOI in July, 1995. The program was funded by the U. S. Office of Naval Research and was aimed at providing technology which can enhance the performance of existing nonacoustic surveillance instrumentation with the ability to incorporate environmental knowledge at the observation location.

2. INSTRUMENTATION

Chip Cox and Dave Jacobs, of SIO, designed and built an instrument to observe electromagnetic fluctuations. The instrument, named Charlotte's Webb, has six radial arms with pressure and electric-field sensors at 0, 1.2, 2.4, 4.8 and 9.8 meters from the center. There are also sensors on the midpoint of the chord between spokes at 4.8 m radius. Two pair of electrodes were mounted vertically, 1 m above the center point. SIO also deployed three moored current meters and conducted CTD casts during deployment and recovery of a small "modem pod".

WHOI designed and built two tripods which recorded temperature, pressure, velocity and vorticity. The tripods were also equipped to measure orientation, pitch and roll of each tripod. The tripods are illustrated in Figures 1 and 2 and described in Tables 1 and 2. The vorticity meters were designed using electronics based on the Benthic Acoustic Stress Sensor (BASS) (Williams et al., 1987).

As illustrated in Figure 1, VORT#2 housed two small-scale vorticity meters (45 cm) to provide horizontal components of velocity at 1.0, 1.4, 3.0 and 3.4 m above bottom, as well as three-dimensional vorticity at 1.2 m and 3.2 m above bottom. VORT#3, as illustrated in Figure 2, housed one large-scale vorticity meter (150 cm), which was built within the structure of the

tripod, providing horizontal components of velocity at 1.1 m and 2.4 m above bottom and vorticity at 1.25 m. For each vorticity meter, the velocity was observed along twelve acoustic paths, defined as u1, u2, u3 ..., u12, as shown in Figure 3a. These paths can be used to compute horizontal velocity (Figure 3b) and vorticity (Figure 3c). The use of each path and the definition of a real-world coordinate system are described in Section 5.

The differential pressure sensors were Paroscientific[1] Model No. 8DP700-1. The manufacturer's calibration coefficients and algorithm are included in Appendix A as seepress.m. On VORT#2, thermistors were potted into the vorticity meter stalks, about 2.5 cm away from the transducer. VORT#3 thermistors were strapped to a leg. The thermistors (YSI[2]) were calibrated at WHOI, as described in Section 5.

Each tripod had been equipped with an Ocean Instrument Systems[3] acoustic release mechanism and a Benthos[4] sphere ("pop-up") housing a hard drive which recorded mirrored data from each data logger using inductive modems. Each sphere was also equipped with an ARGOS[5] transmitter to assist in locating the pop-up upon release.

1.Paroscientific, Inc. Redmond, WA 98052

2.Yellow Spring Instruments.

3.Ocean Instrument Systems, North Falmouth, MA 02556

4.Benthos, Inc., North Falmouth, MA 02556

5.ARGOS receiver/transmitters are manufactured by NACLS, INC., Landover, MD 20785

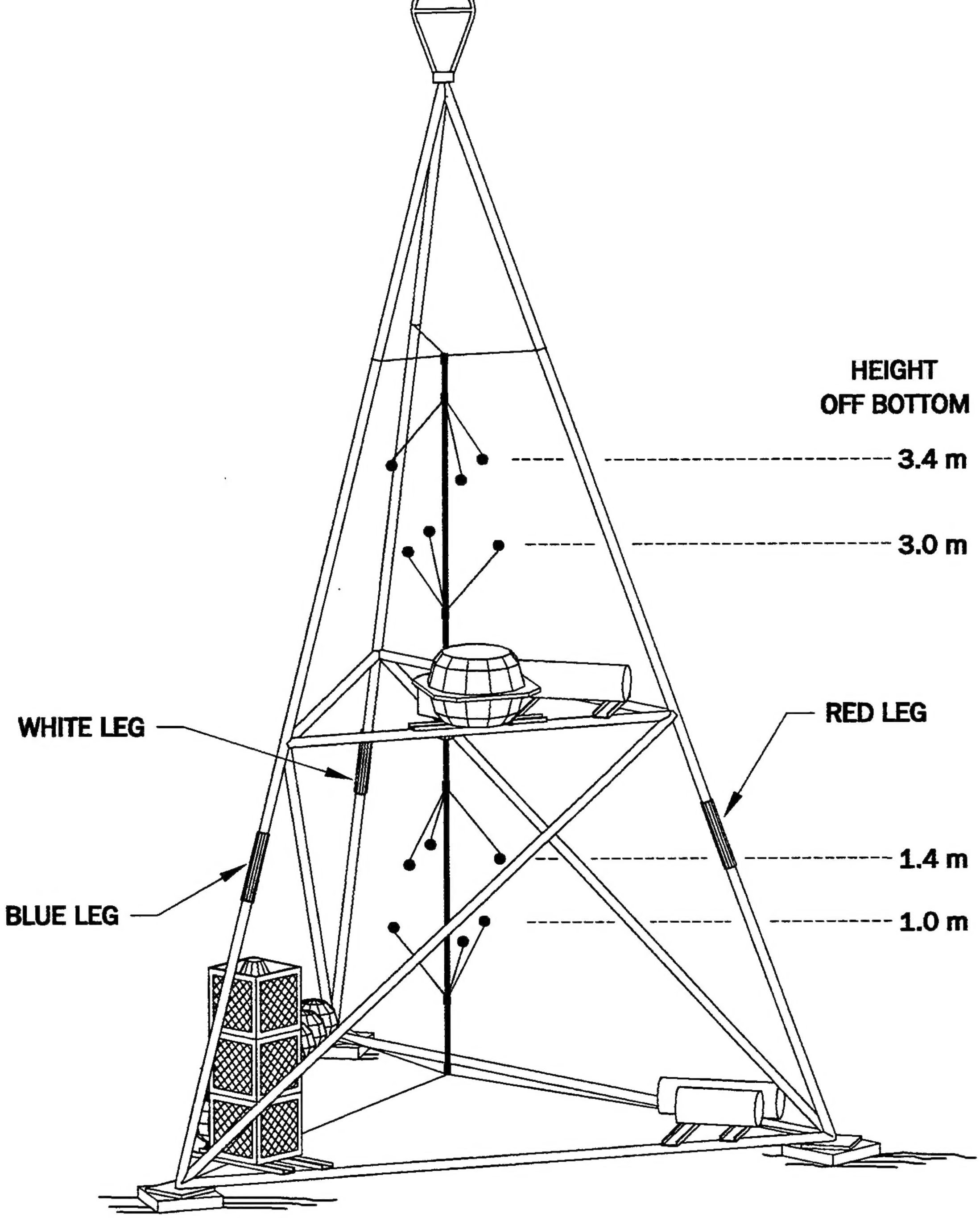


Figure 1. VORT#2 small scale vorticity meter, showing the heights of the horizontal velocity planes. The length of each acoustic path is 45.

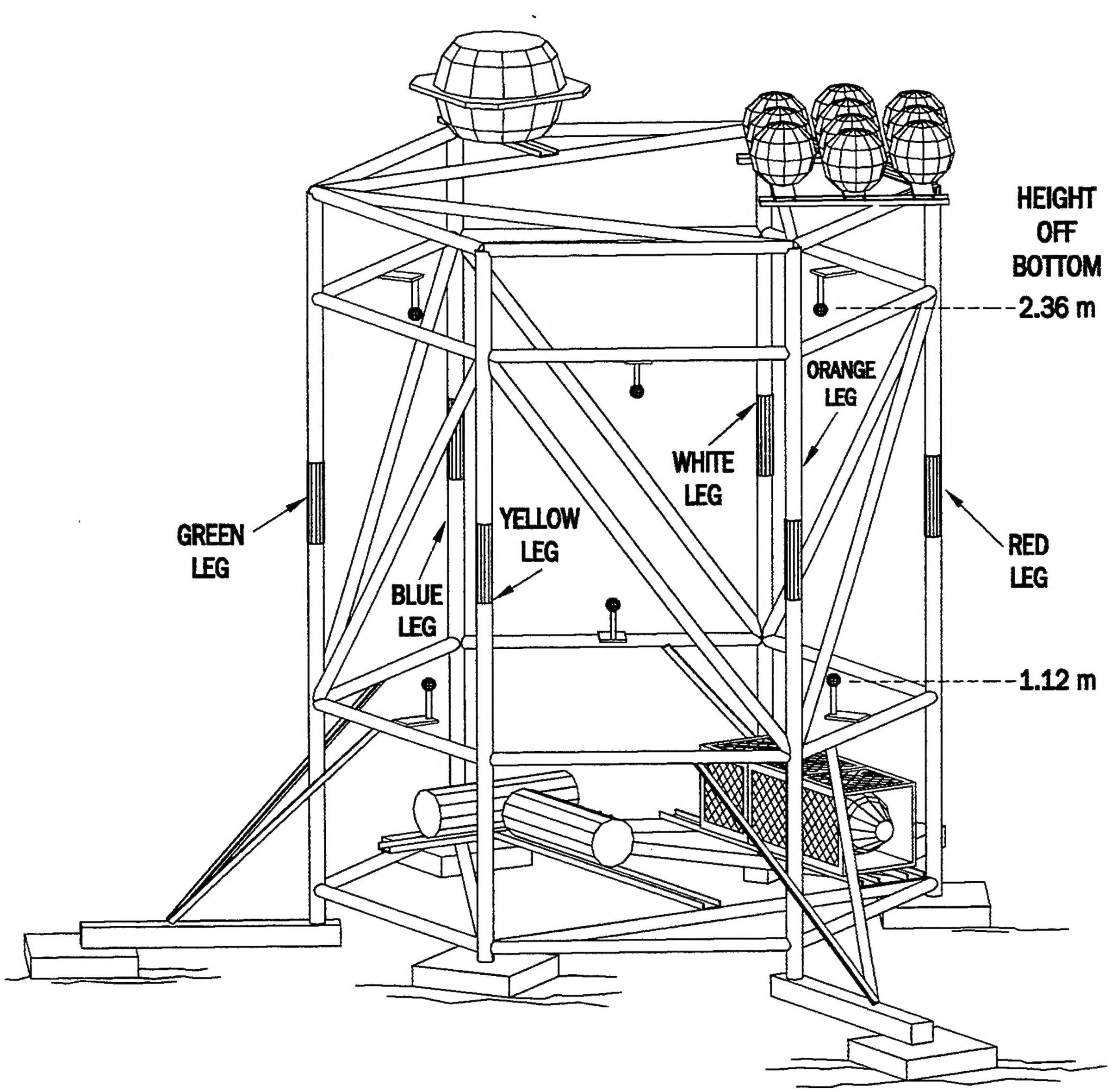


Figure 2. VORT#3 large scale vorticity meter, showing the heights of the horizontal velocity planes. The length of each acoustic path is 150 cm.

Table 1. VORT#2 Instrumentation		
Measurement	*Height Above Bottom (cm)*	*Description*
Thermistor#1	104	
Velocity	105	Bottom Sensor - Bottom Plane (Paths 1,5,9)
Vorticity	125	(Center of Each Plane)
Velocity	145	Bottom Sensor - Top Plane (Paths 3,7,11)
Thermistor#2	149	
Pressure	227	Paroscientific SN57216
Thermistor#3	304	
Velocity	308	Top Sensor - Bottom Plane (Paths 13,17,21)
Vorticity	337	(Center of Each Plane)
Velocity	350	Top Sensor - Top Plane (Paths 15,19,23)
Thermistor#4	354	

Table 2. VORT#3 Instrumentation		
Measurement	*Height Above Bottom (cm)*	*Description*
Pressure	33	Paroscientific Inst SN57222
Thermistor#1	36	
Thermistor#2	59	
Velocity	112	Bottom-Plane (Paths 1,5,9)
Thermistor#3	112	
Thermistor#4	139	
Thermistor#5	167	
Vorticity	175	(Center of Each Plane)
Thermistor#6	210	
Velocity	240	Top-Plane (Paths 3,7,11)
Thermistor#7	251	
Thermistor#8	297	

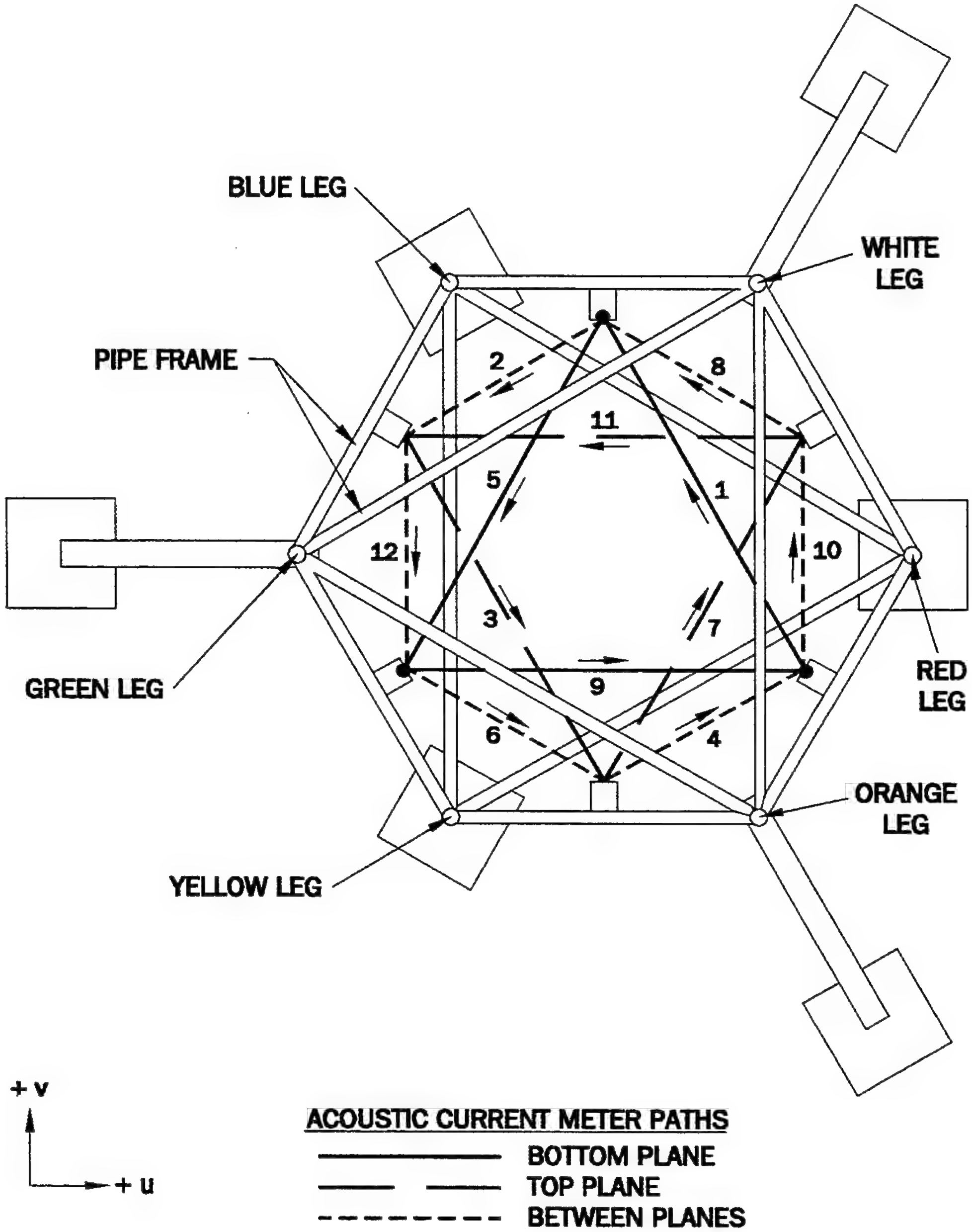


Figure 3a. Definition of each acoustic path for VORT#3. The explanation of u and v can be found in Section 5.

Figure 3b.

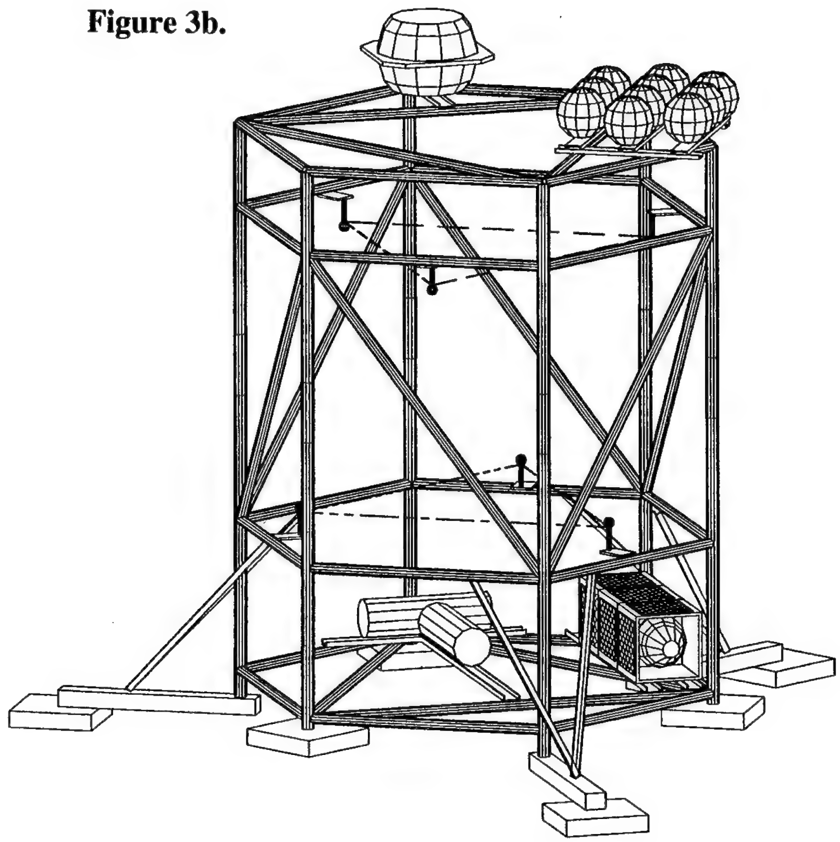
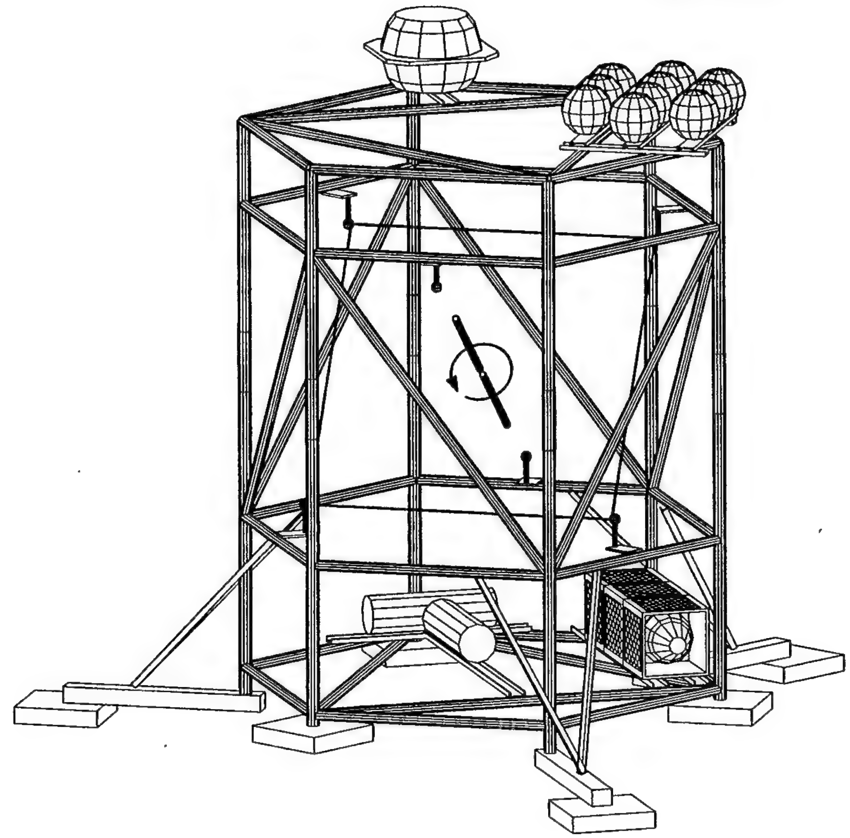


Figure 3c.



Figures 3b & 3c illustrate the horizontal velocity planes and an example of a vorticity plane, respectively.

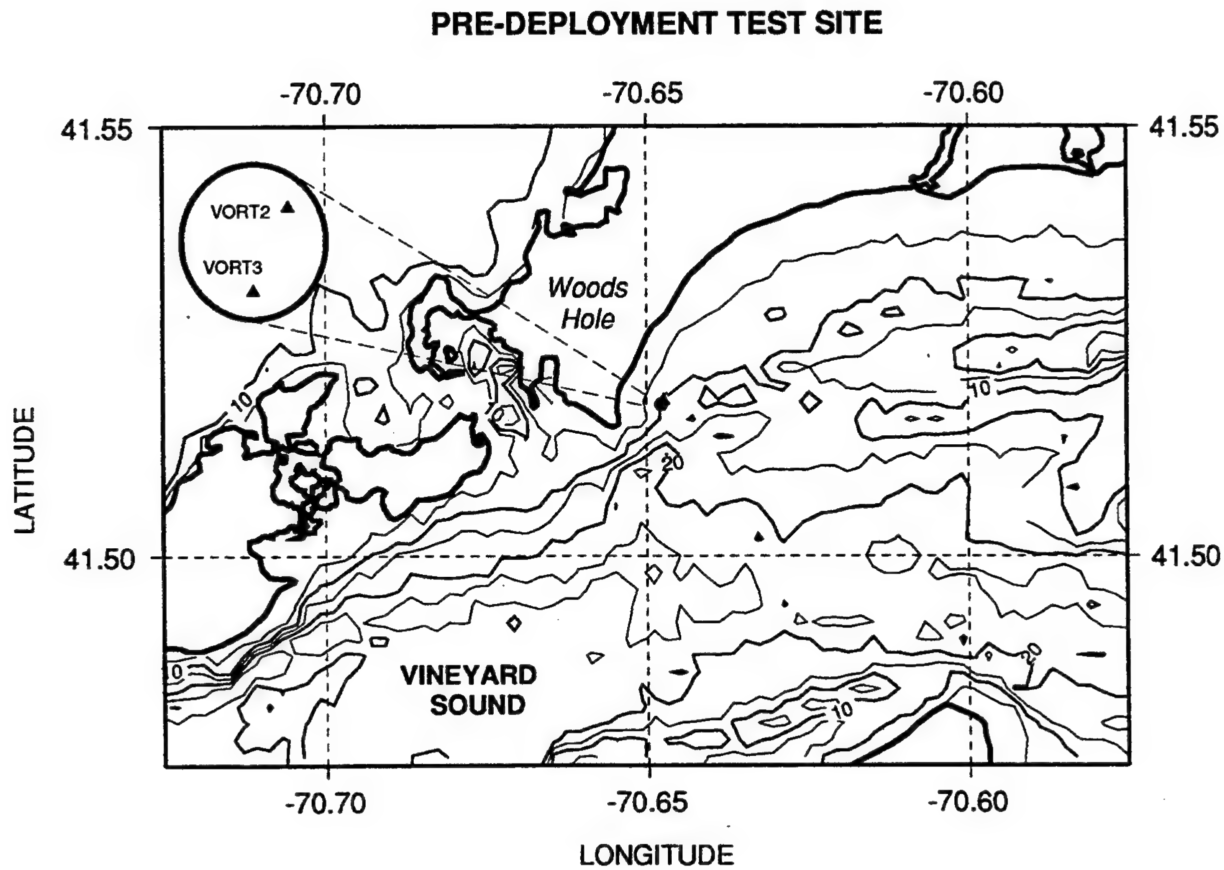


Figure 4. Bathymetric map shows depth contours (in meters) with location of pre-deployment test sites. The tidal excursion was estimated to be approximately 5 km, based on tidal velocities of 0.75 m/s. VORT#2 and VORT#3 were positioned approximately 60 meters apart, along the 10 meter isobath.

3. PRE-DEPLOYMENT TESTS

WHOI conducted three deployments to test the newly developed tripods and instrumentation. The tripods were placed approximately 200 feet apart about a quarter-mile east of Nobska Point (Woods Hole, MA) in Vineyard Sound along the 10 meter isobath. (See Figure 4.) The tests proceeded as follows:

May 7-10 1994 - to observe instruments for strumming in high currents (no electronics)

May 28-30, 1994 - with electronics continuously sampling at 4 Hz for less than one day

June 17-22, 1994 - with electronics sampling at 7.14 Hz for 72 seconds, every other minute for 34 minutes every third hour for five days

The May 7th deployment indicated that no significant strumming occurred, as observed by three divers during peak tide, which is presumed to be approximately 1 m/s (based on subsequent measurements).

During the May 28th deployment, continuous data were collected until the disk was full, which occurred after approximately 14 hours. It appears that the VORT#2 tripod tipped over during strong flow. Therefore, there were less than 2 hours of data for the small scale vorticity meter. Upon analysis of the pressure data from VORT#3, we learned that a clock interrupt was not being serviced, effecting a clock cycle of 280 ms, not 250 ms, as planned. This affects the pressure measurement and the frequency spectrum (i.e., 3.57 Hz sampling, not 4Hz). The VORT#3 electronics had been set to ±0.75 m/s for maximum count and, therefore, velocity buffers overflowed at peak current. These problems have subsequently been corrected. The internal clocks started when power was supplied, causing a 0.5165 day lead in the VORT#2 clock.

The thermistors were calibrated at the WHOI CTD Calibration Lab on June 9-10. Coefficients for conversion to temperature are listed in Appendix A, with the MATLAB®[6] script used for the conversion. The sampling scheme was similar to that actually used in the Strait of Juan De Fuca. VORT#2 and VORT#3 dataloggers were turned on June 17th at 11:30 and 11:45 EDT, respectively. The tripods were deployed at 12:16 and 12:18 EDT, respectively. No data from the VORT#2 tripod were recorded, due to an electronics failure. The acoustic paths on VORT#3 were atypically noisy. This noise is reasonably explained as being attributable to seaweed. The tripod was covered with it upon recovery and there appears to be more noise in the top-most plane, where the acoustic paths are below tubing which trapped seaweed.

6.MATLAB is a product of The Mathworks, Inc., Natick, MA 01760

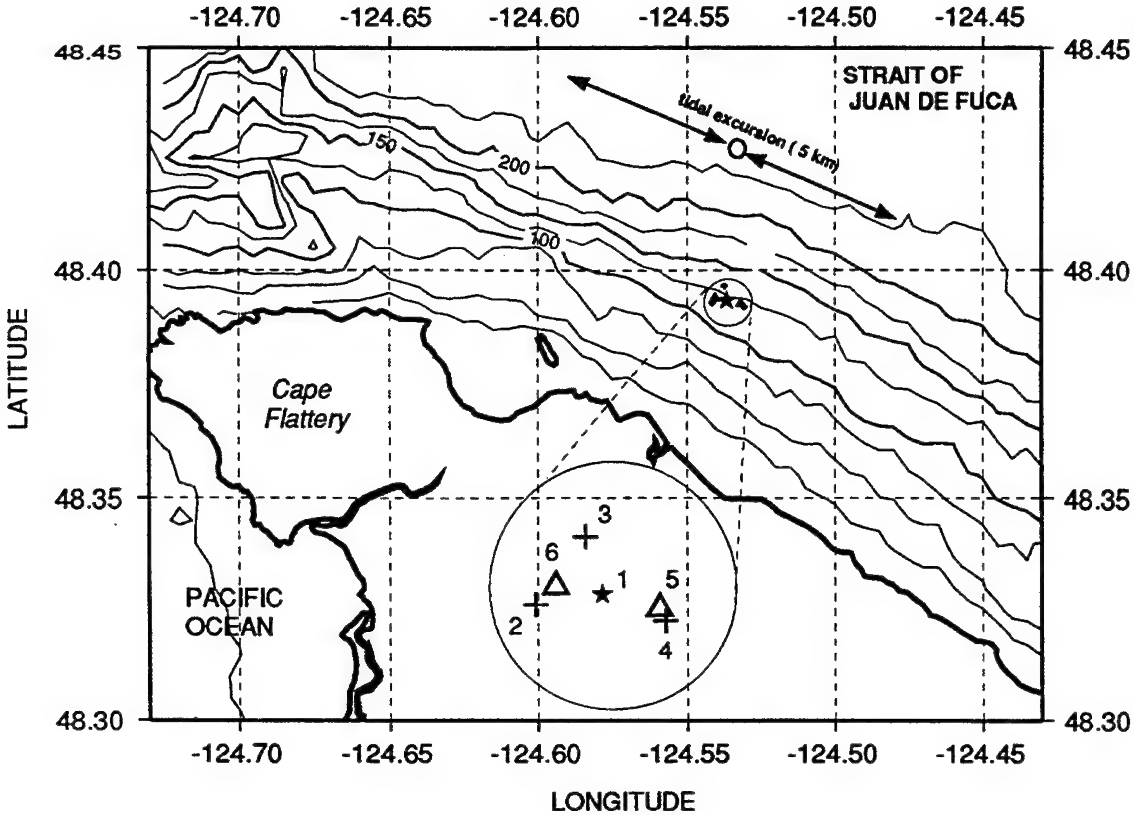


Figure 5. Bathymetric map shows depth contours (in meters) with location of deployment sites. The tidal excursion was estimated to be approximately 5 km, based on tidal velocities of 0.75 m/s. VORT#2 and VORT#3 were positioned approximately 600 meters apart, along the 125 meter isobath. The positive along-strait velocity (u) is defined as flow along the isobath (positive is eastward). The cross-strait velocity (v) is defined as flow shore-normal flow (positive is northward).

Table 3. Instrument Summary

Map Site / Organization	*Instrument Name*	*Property Observed*	*Latitude (North)*	*Longitude (West)*	*Deployment (GMT)*
1/SIO	Charlotte's Webb	Electro-magnetic Fluctuations	48° 23.59'	124° 32.19'	~7/11/94 15:00
2/SIO	M1	CTD/VMCM	48°23.56'	124° 32.49'	
3/SIO	M2	CTD/VMCM	48° 23.79'	124° 32.24'	
4/SIO	M3	CTD/VMCM	48° 23.51'	124° 31.84'	
5/WHOI	VORT#2	Vorticity Temperature Pressure	48° 23.55'	124° 31.91'	7/11/94 22:48
6/WHOI	VORT#3	Vorticity Temperature Pressure	48° 23.62'	124° 32.43'	7/11/94 20:36

4. DEPLOYMENT & RECOVERY

The tripods were shipped to the University of Washington (UW), where they were reassembled and loaded onto the Research Vessel Thomas G. Thompson and deployed with the SIO instruments in the Strait of Juan De Fuca, as shown in Figure 5.

The VORT#2 sensors were placed in barrels of stationary, unstratified water to obtain pre-deployment zeros, which were recorded for subsequent processing. The VORT#3 tripod was tested from the ship, while it was docked at UW. No zeros can be measured for the VORT#3 tripod due to its size.

The clocks for VORT#2 and VORT#3 were started on July 11, 1994, at 18:04 and 18:06 GMT, respectively. The SIO CTD casts were conducted and the SIO current meters were deployed after Charlotte's Web but before VORT#3. VORT#2 had been deployed prior to VORT#3 (at 19:42 and 19:55) but had to be recovered when the recovery floats released from the strain of being dragged on deployment. After a stronger bolt secured the release mechanism, VORT#2 was redeployed at the position and time listed in Table 3. The depth at the site was noted as 125 meters, from the ship's fathometer. The GPS positions noted in Table 3 have inherent inaccuracies on the order of 50-100 meters.

On August 23, the pop-ups were triggered and recovered aboard a vessel (Satin Doll) chartered from Far West Resorts. The data were recovered for the large scale vorticity meter (VORT#3) and transferred for processing. It became apparent at this time that VORT#2 had failed, although it had successfully logged the zeros the day before deployment.

The logger was filled by March 27, 1997, and the instruments were recovered aboard the SIO R/V Robert Gordon Sproul, on May 28, 1997.

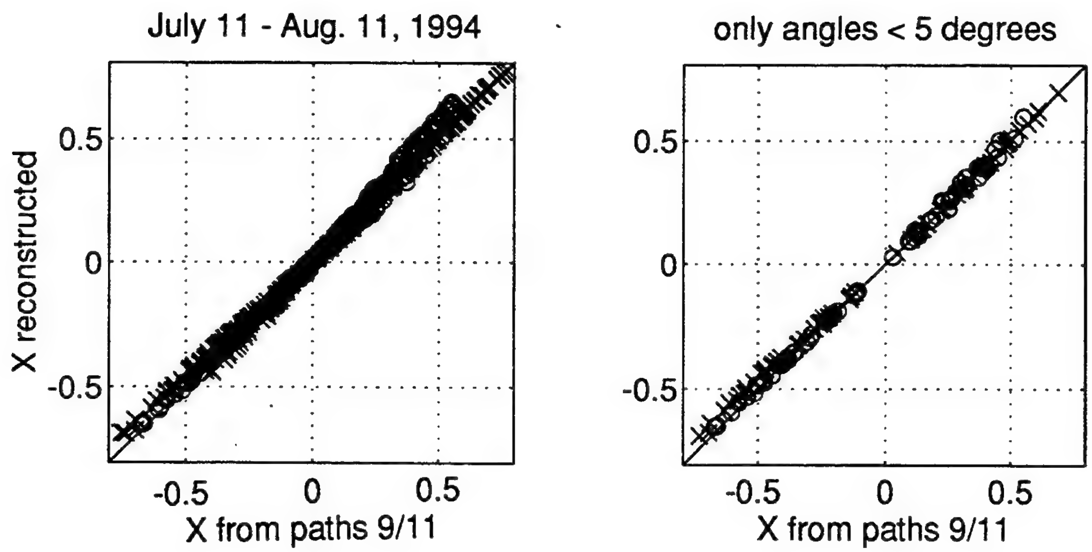


Figure 6a & 6b. Comparison of along-strait velocity components from paths 9 (o) and 11 (x) with each corresponding reconstructed along-strait velocity.

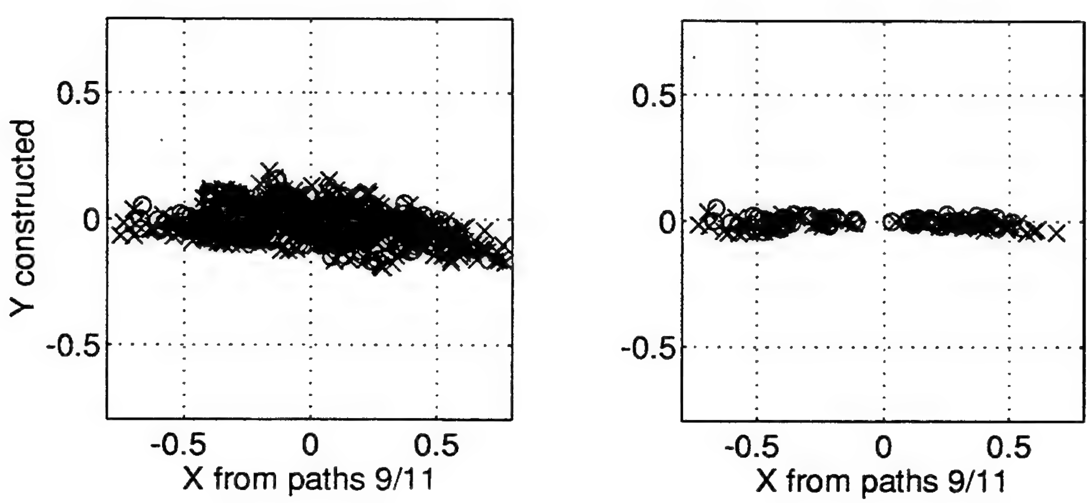


Figure 6c & 6d. Comparison of cross-strait velocity components with along-strait velocity components in each velocity plane.

5. DATA PROCESSING & ANALYSIS

VORT#3 data were unpacked and stored in files of daily records with one variable per file. Each continuous time series consists of 514 samples (72 seconds $\times$ 7.14Hz), measured every other minute (seventeen times), providing approximately 34 minutes of sampling (514 $\times$ 17 or 8738 records): 72 seconds on / 48 seconds off. These observations were repeated every other hour for 24 hours (104856 records) every third day for a month. This sampling was repeated every third month until the disk was full, which occurred in March 1995. The compass heading, pitch and roll remained consistent at 332°N, -2° and 4°, respectively. The unpacking and storage routine, bin2mat.c, is included in Appendix C with the TTBASIC® programs which controlled the sampling.

A coordinate system is defined so that x is along-strait (positive in the flood direction), y is cross-strait (positive is poleward, $+21°N_{true}$), and z is vertical (positive upward). The corresponding velocity components are u, v and w, respectively. It can be shown that

$$u_{bot} = u9,$$
$$u_{top} = -u11,$$

where subscript bot and top refer to the horizontal planes, as defined in Table 2 and illustrated in Figure 3b, and uN represents the acoustic path as defined in Figure 3a.

The cross-strait velocity (v) were computed as follows:

$$v_{bot} = (u1-u5)/(2*\cos(30°)),$$
$$v_{top} = (u7-u3)/(2*\cos(30°)),$$

The combination of the other two horizontal paths in each plane provides a consistency check:

$$uc_{bot} = -(u1+u5),$$
$$uc_{top} = (u7+u3),$$

Comparisons of the along-strait velocities (u) with the reconstructed along-strait velocities (uc) are presented in Figures 6a and 6b. Figures 6c and 6d show the cross-strait velocity (v) with the along-strait velocity. When the difference between the observed velocity and the reconstructed velocity are plotted with the angle of flow (Figure 7a and 7b), it appears that there is a flow disturbance when the direction of flow deviates by greater than 5 degrees from the u-path direction. This disturbance is also apparent in the estimates of drag coefficient from mean velocity (u) log-profiles, as seen in Figures 8a and 8b. For this reason, we have restricted some of the analyses to those periods of undisturbed flow. A more detailed discussion of flow error in the vorticity meter can be found in Thwaites (1995).

According to the Kolmogorov model (Batchelor, 1953), the spectral density within the inertial subrange can be used to estimate the dissipation rate as follows:

$$S(k) = \beta\alpha\varepsilon^{2/3}k^{-5/3}$$

where the constant β depends on whether the spectral estimate is determined from the along-path flow (which is the case for paths 9 and 11) or from the across-path flow (paths 10 and 12) and is defined as follows:

$$\beta = 9/55 \text{ for paths 9 and 11,}$$

and

$$\beta = 12/55 \text{ for paths 10 and 12.}$$

As seen in Figures 9 - 11, filtering of the turbulence is exhibited when the flow is in the direction of the acoustic path. Although this effect is not consistent with quantitative calculations (Kaimal et al., 1968), it appears to be repeatable. To avoid influence of the bottom and contribution from aliasing, wave numbers for the estimate were restricted to $kz > 5$ and $kz < k_s/10$, where

$$k_s = 2\pi Fs/u.$$

and Fs is the sampling frequency. Therefore, we assumed an along-path filter proportional to 1/k and computed an estimate of a dissipation-like quantity as:

$$A_{u9,u11} = \Sigma(S(k)\cdot k^{5/3}\cdot k)\ /N_k = \Sigma(S(k)\cdot k^{8/3})/N_k$$

evaluated from $kz > 5$ and $kz < k_s/10$.

It can be shown that the vortex shedding in the cross-flow paths (u10/u12), is consistent with tubing diameters of the tripod structure and also coincides with flow direction, i.e. path u10 is contaminated strongly during strong ebb and u12 is most affected during strong flood. For cross-path flow, u10/u12, spectral density analysis is restricted to regions above $kz = 5$, and is further restricted by the apparent vortex shedding to regions below $kz = 12$. For the cross-path estimates,

$$A_{u10,u12} = \Sigma(S(k)\cdot k^{5/3})/N_k.$$

As well as using only periods of undisturbed flow, for the estimates of dissipation, these data sets were further restricted to periods of stationary flow. The reverse arrangements test, as described in Bendat (1986), determined stationarity in the mean and standard deviation of the mean velocity, u, at 2.4 m above bottom. Summaries of the estimates of A are included as Figures 12a and 12b.

To compute vorticity, the four acoustic paths which form a closed square are summed and divided by the length of each path (1.5 for the large scale vorticity meter, VORT#3):

$$V_A = (u1+u2+u3+u4)/L;$$

$$V_B = (u5+u6+u7+u8)/L;$$

$$V_C = (u9+u10+u11+u12)/L;$$

The vorticity at each sample were rotated into the x, y, z coordinate system as described in Appendix B. The 48 seconds of unobserved vorticity were included as zeros and spectral analysis of the vorticity was computed using 4 minute windows. July spectra are shown in Figure 13.

An estimate of the variance in vorticity in a given observation was determined by

$$\int_{kz>0.1}^{kz<1} S(k) \bullet 2dk$$

Figure 14a and 14b confirm the nonconformity of ebb and flood conditions in vorticity, as seen in the velocity and dissipation analyses.

Figure 7a.

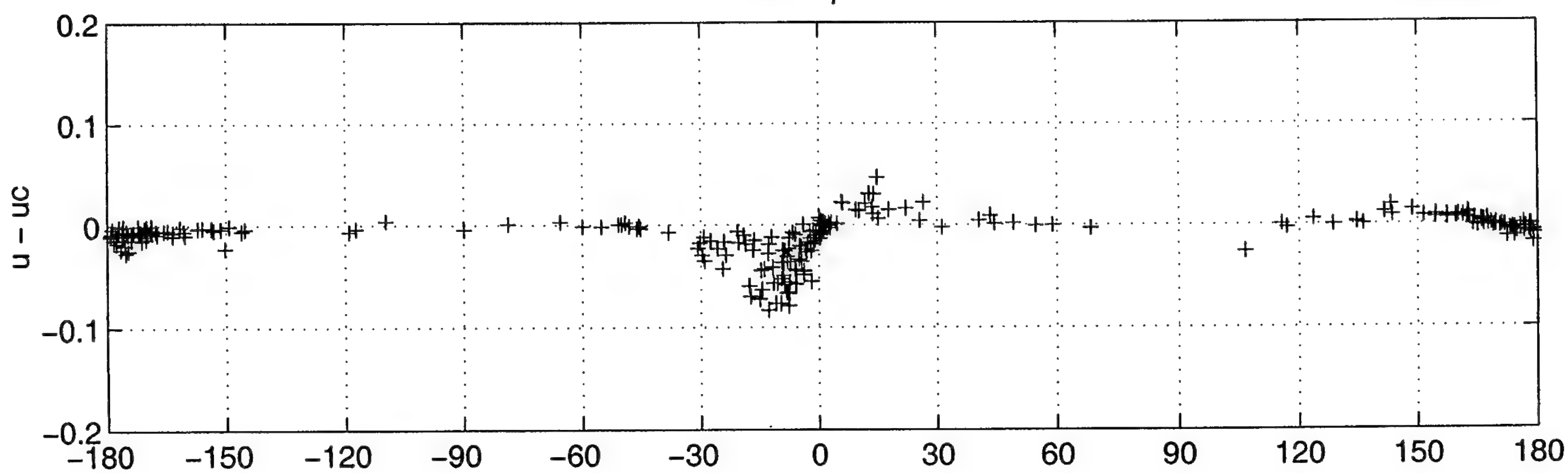


Figure 7b.

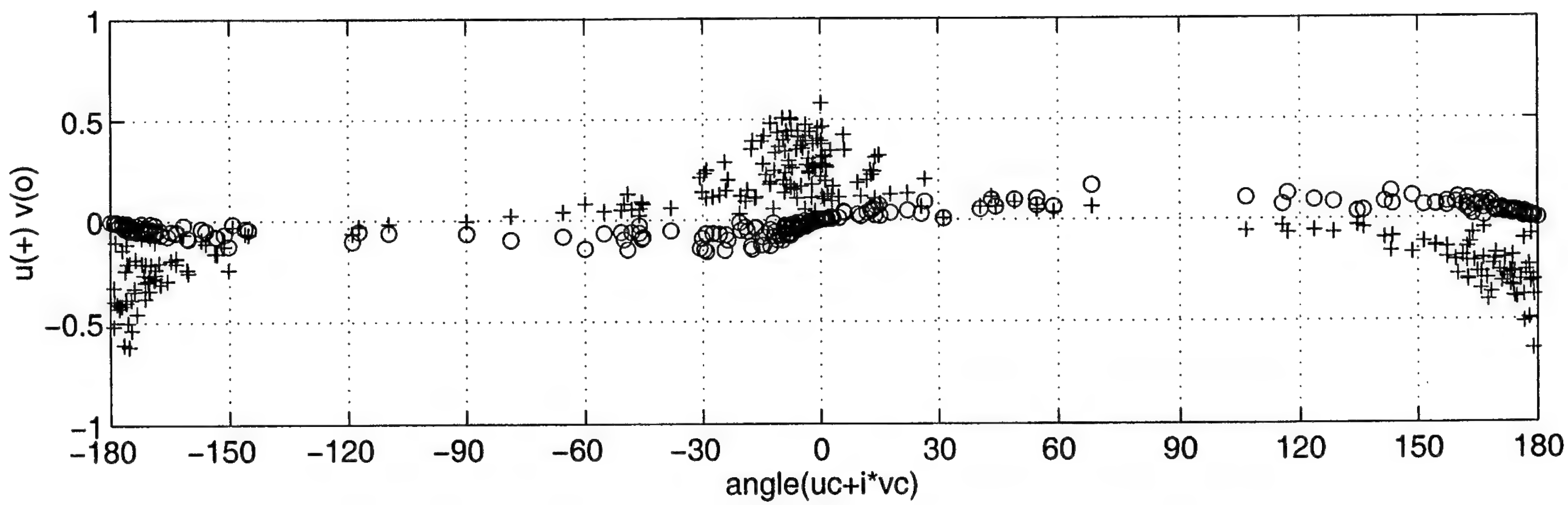


Figure 7c.

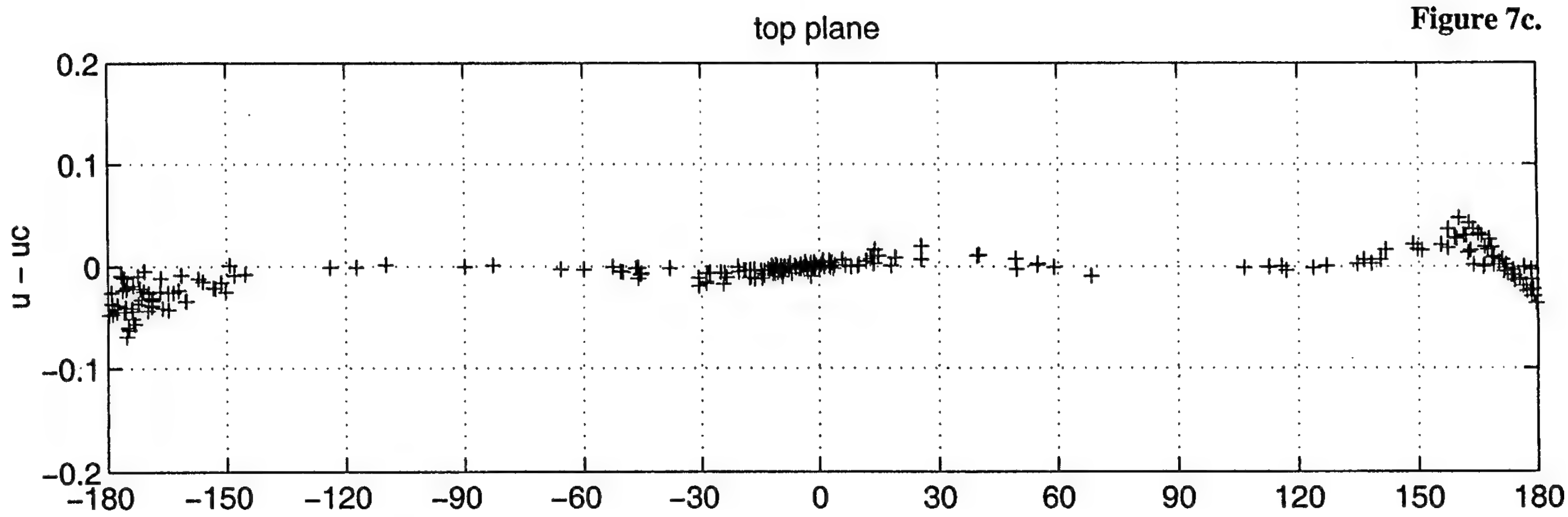


Figure 7d.

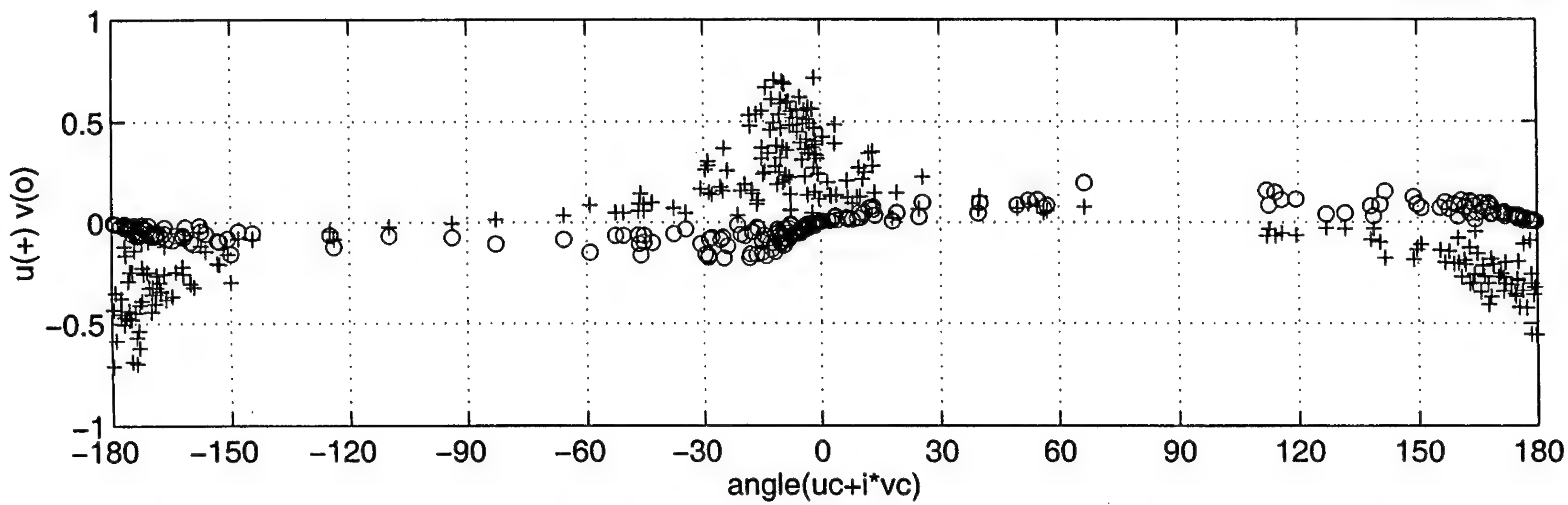


Figure 7. Figures 7b & 7d illustrate the amplitude of the along-strait (u) and cross-strait (v) velocity. Figures 7a & 7c document the difference between the along-path flow (u) and the reconstructed along-strait flow (uc), at the bottom plane and top plane, respectively.

Figure 8a.

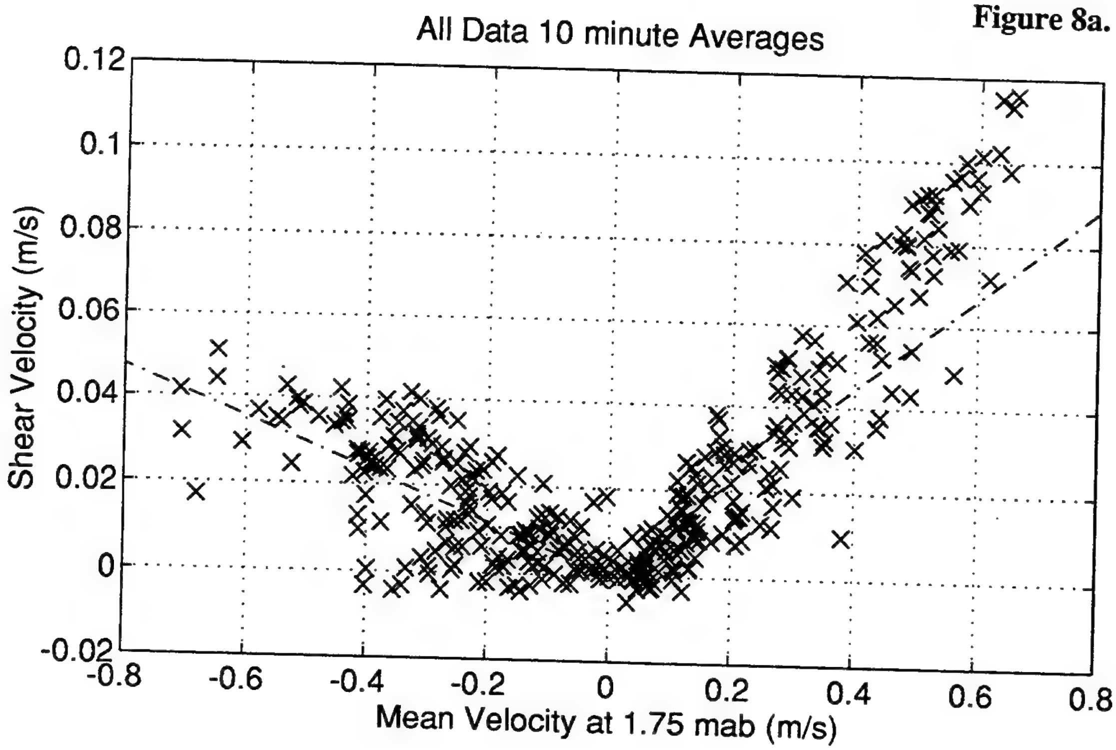


Figure 8b.

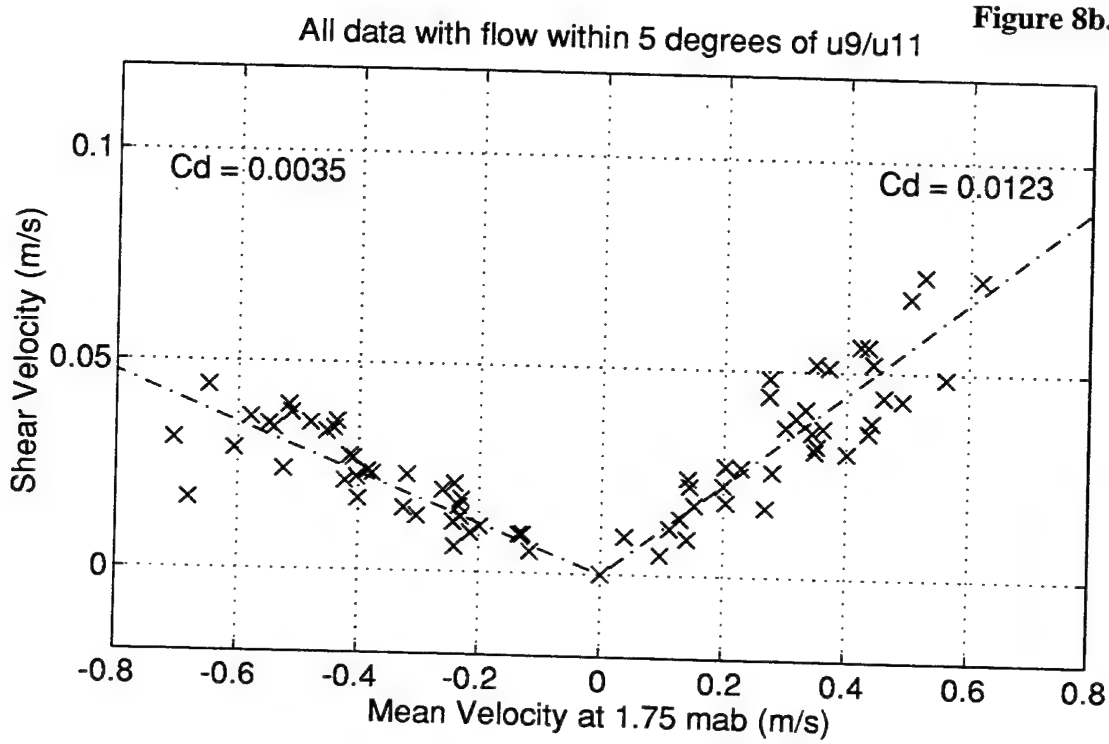
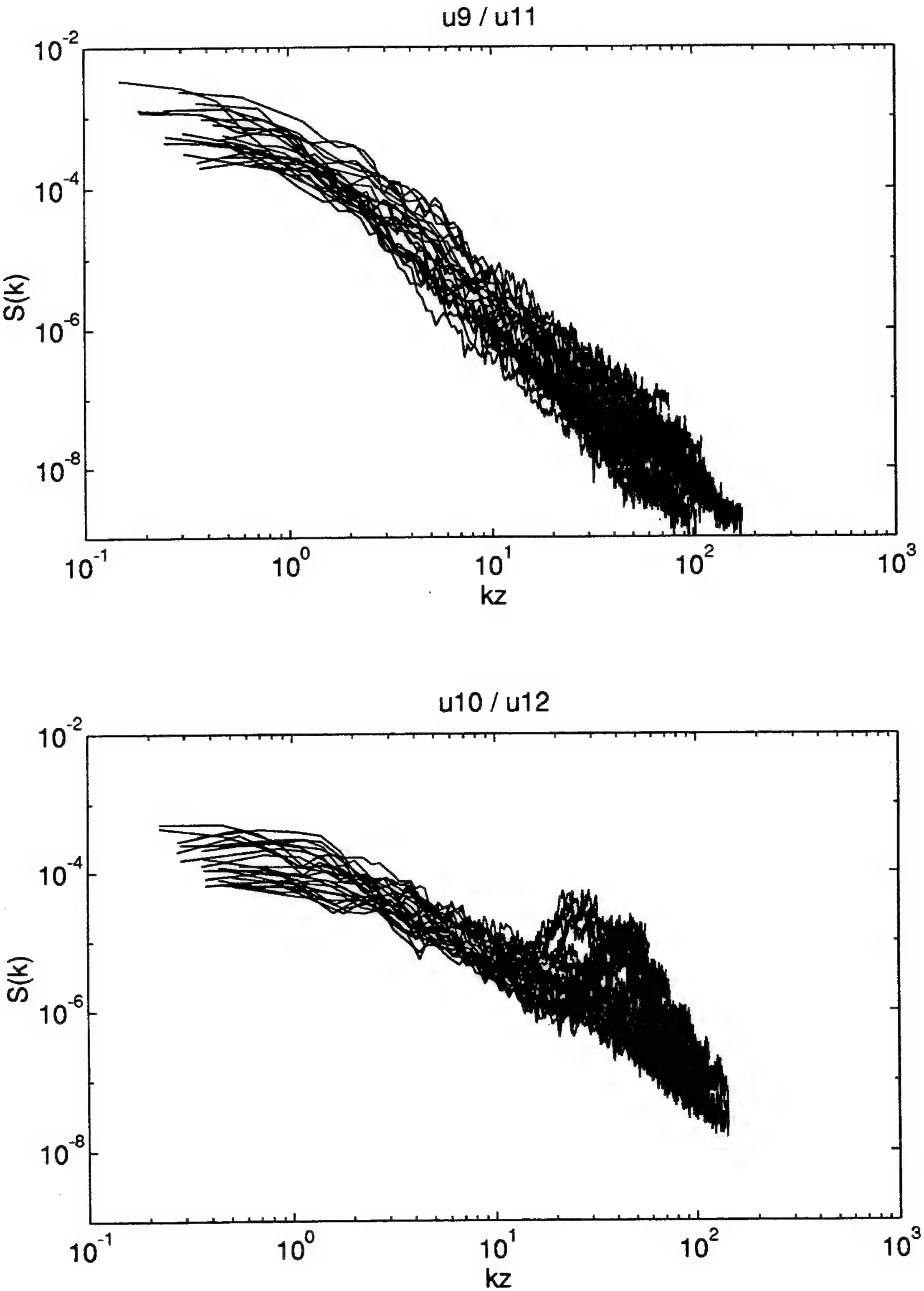


Figure 8.



u9 / u11
S(k)
10-2
10-4
10-6
10-8
10-1
100
101
102
103
kz
u10 / u12
S(k)
10-2
10-4
10-6
10-8
10-1
100
101
102
103
kz

Figure 9.

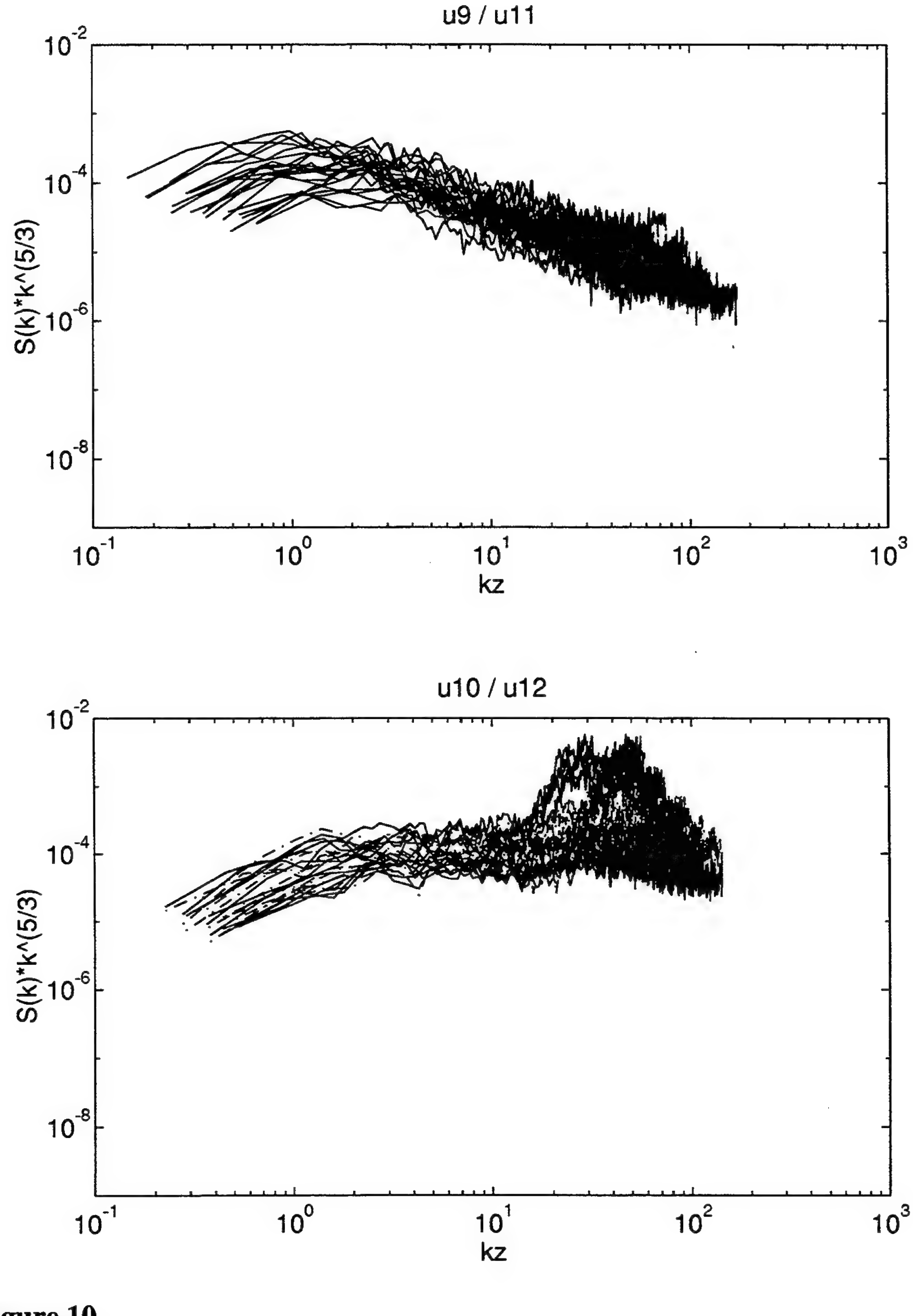


Figure 10.

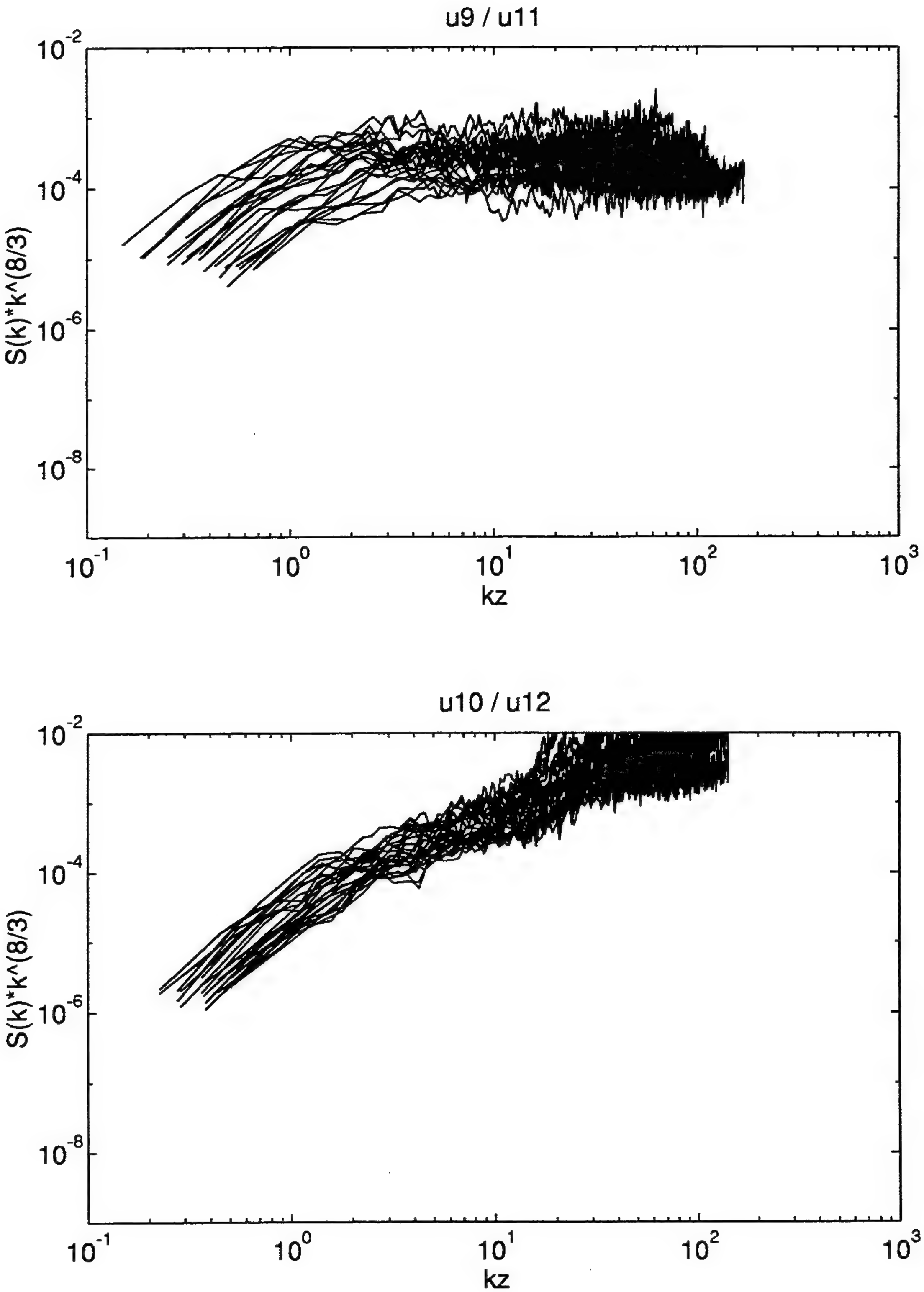


Figure 11.

Figure 12a. u9(bottom-o) & u11(top-x)
(dashed line represents a slope of $9/55\alpha C_d$)

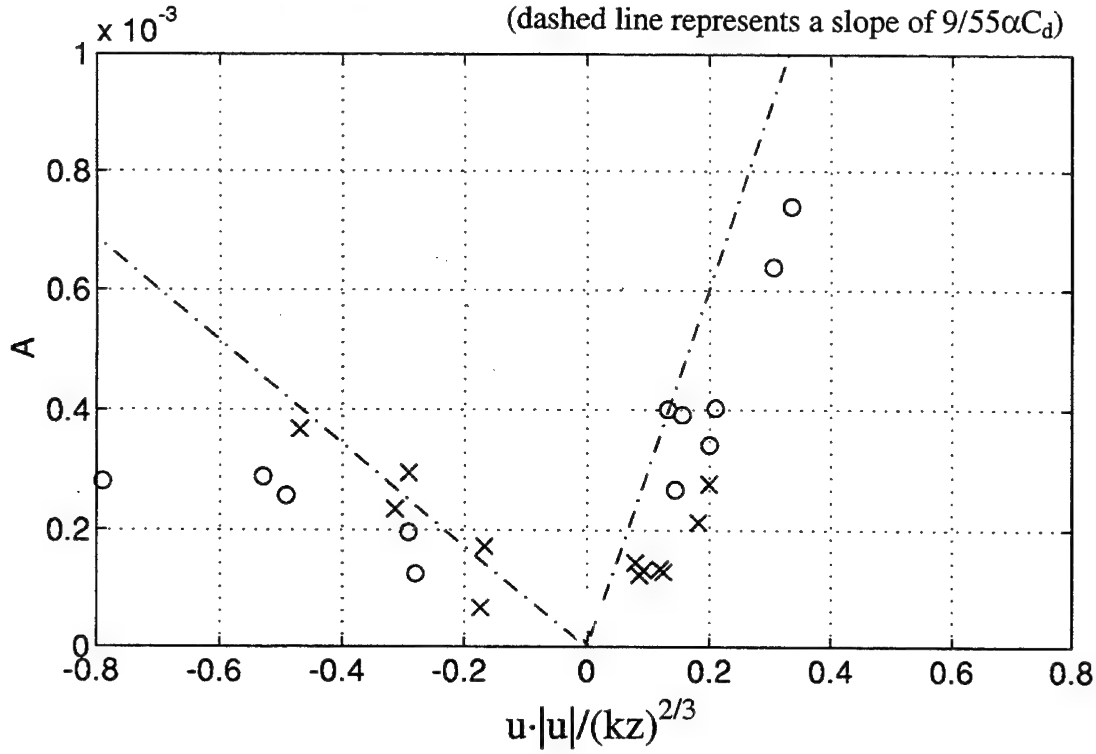


Figure 12b. u10/u12 (dashed line represents a slope of $12/55\alpha C_d$)

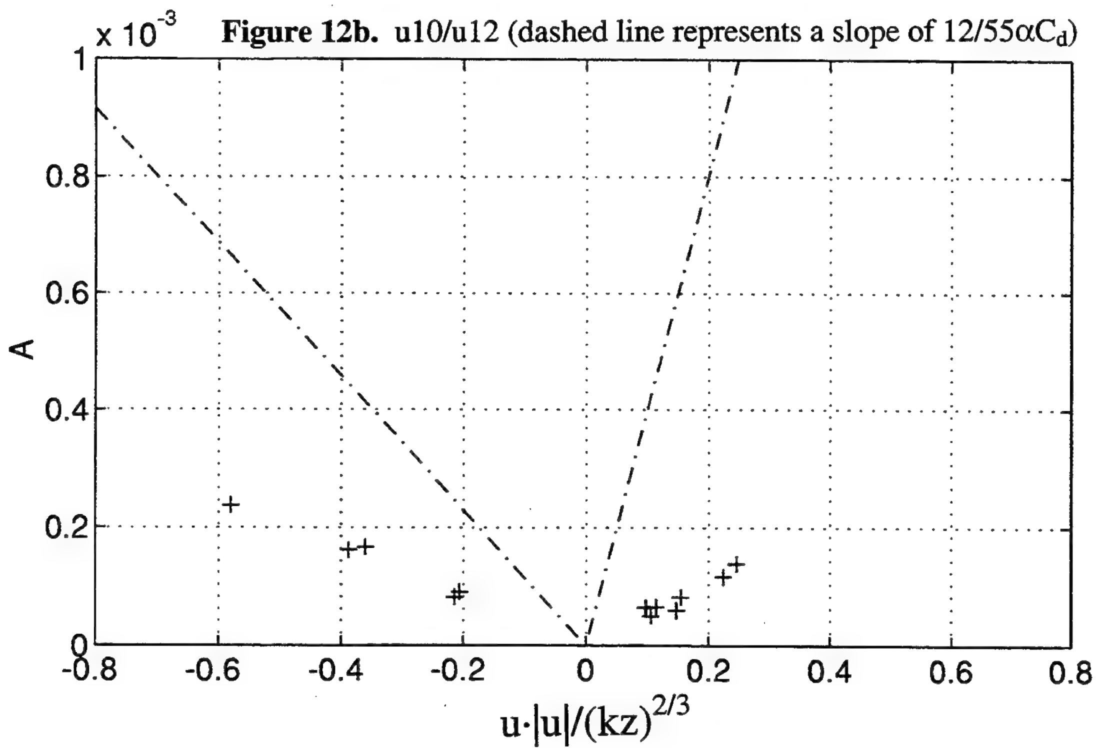


Figure 12. $\alpha = 1.5$ and, based on Figure 8b, $C_d(\text{flood}) = 0.0123$, $C_d(\text{ebb}) = 0.0035 u \cdot |u|/(kz)^{2/3}$

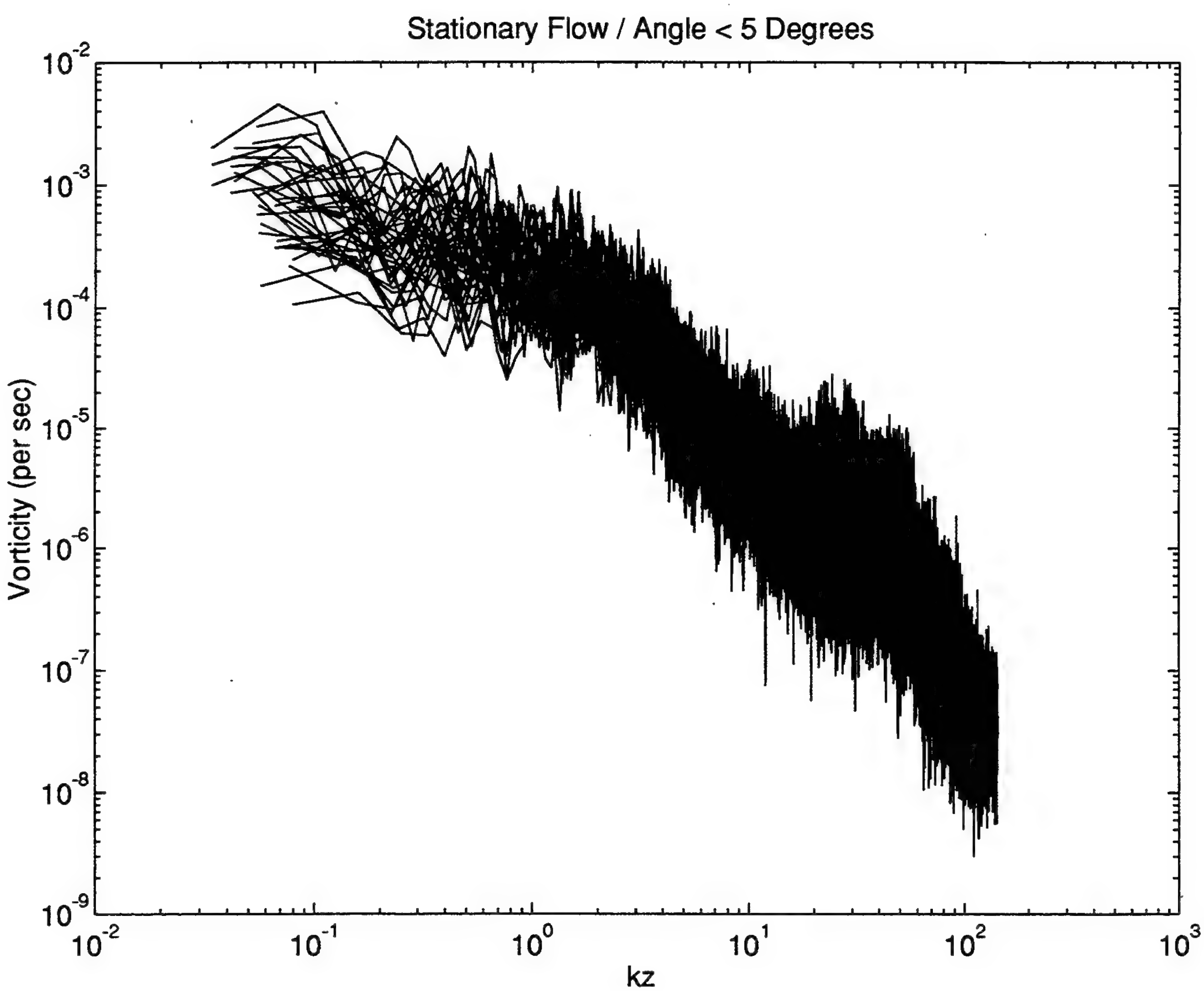

Stationary Flow / Angle < 5 Degrees
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
10^{-8}
10^{-9}
Vorticity (per sec)
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
kz

Figure 13.

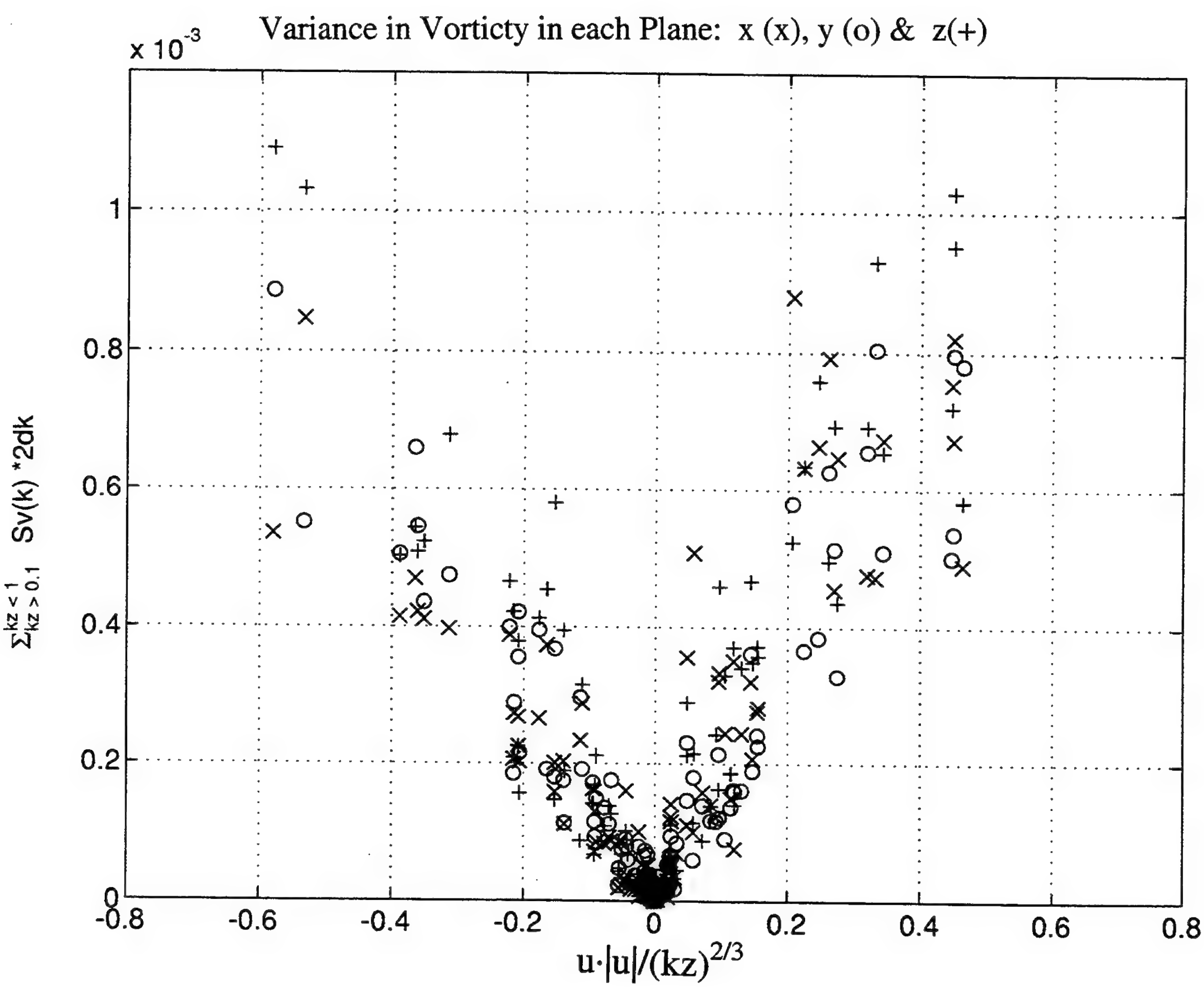

Variance in Vorticty in each Plane: x (x), y (o) & z(+)
x 10^-3
1
0.8
0.6
0.4
0.2
0
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
u·|u|/(kz)^2/3
Σ kz < 1 kz > 0.1 Sv(k) *2dk

Figure 14.

6. DATA SUMMARIES

Figures 15 - 24 provide time series of the horizontal components of velocity, along with the temperature in the near-bottom flow and the large scale vorticity variance in wave numbers ($0.1 < kz < 1$).

The velocities presented in Figures 15 - 19 represent 72 second means which were sampled every other minute seventeen times (~ 33 minutes) every other hour. The temperatures include data from thermistors 1, 2, 3,4,5,6 and 8. They are included here to document the absence of stratification. The thermistors present data from 0.33 meters above bottom to 2.97 meters above bottom. The standard deviation in each 72 second record are offset and plotted with the means.

Vorticity variance within each 72 second burst, as defined in Section 5, is included as Figures 20 - 24.

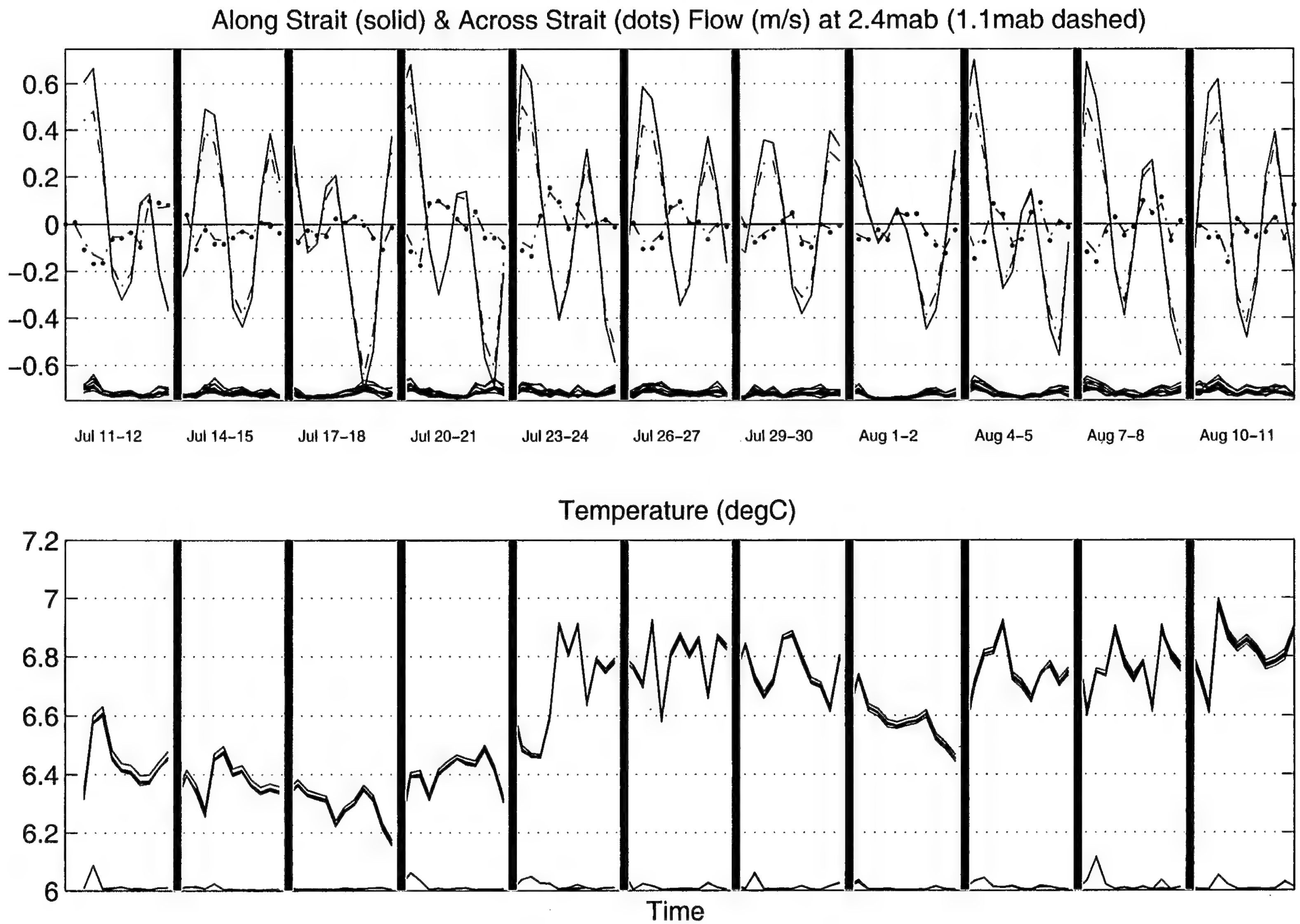

Along Strait (solid) & Across Strait (dots) Flow (m/s) at 2.4mab (1.1mab dashed)
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
Jul 11−12
Jul 14−15
Jul 17−18
Jul 20−21
Jul 23−24
Jul 26−27
Jul 29−30
Aug 1−2
Aug 4−5
Aug 7−8
Aug 10−11
Temperature (degC)
7.2
7
6.8
6.6
6.4
6.2
6
Time

Figure 15.

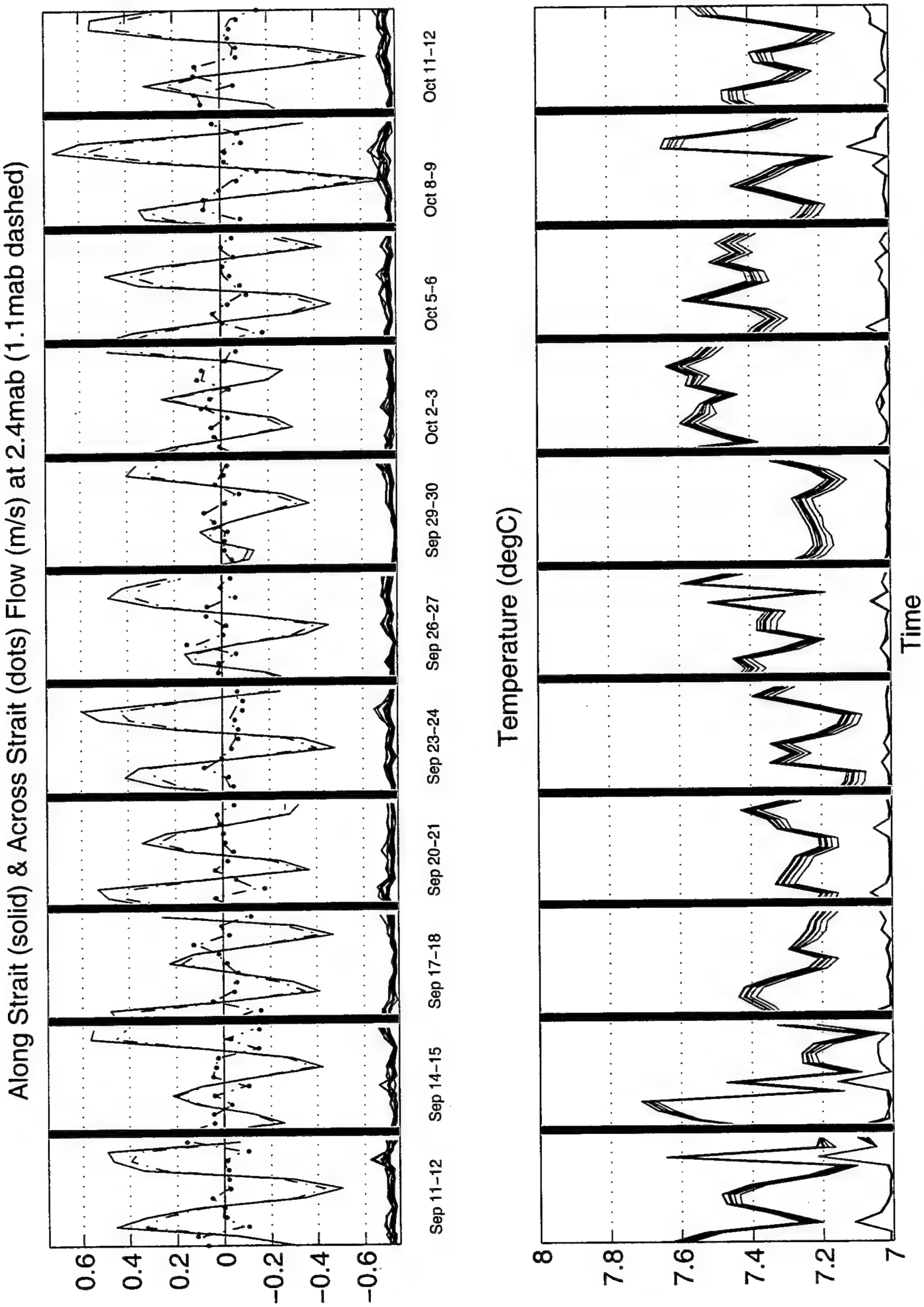

Along Strait (solid) & Across Strait (dots) Flow (m/s) at 2.4mab (1.1mab dashed)
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
Sep 11-12
Sep 14-15
Sep 17-18
Sep 20-21
Sep 23-24
Sep 26-27
Sep 29-30
Oct 2-3
Oct 5-6
Oct 8-9
Oct 11-12
Temperature (degC)
8
7.8
7.6
7.4
7.2
7
Time

Figure 16.

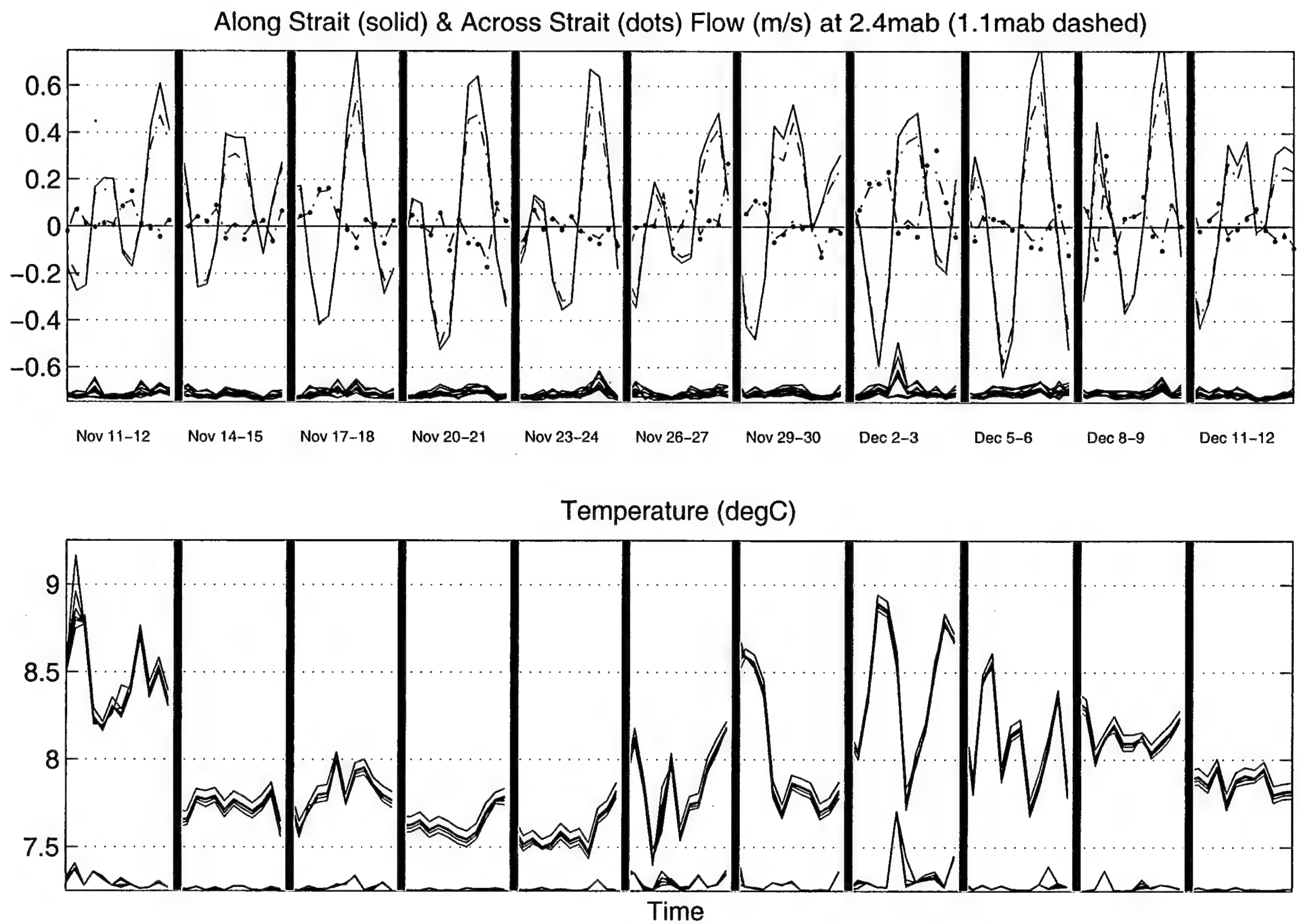
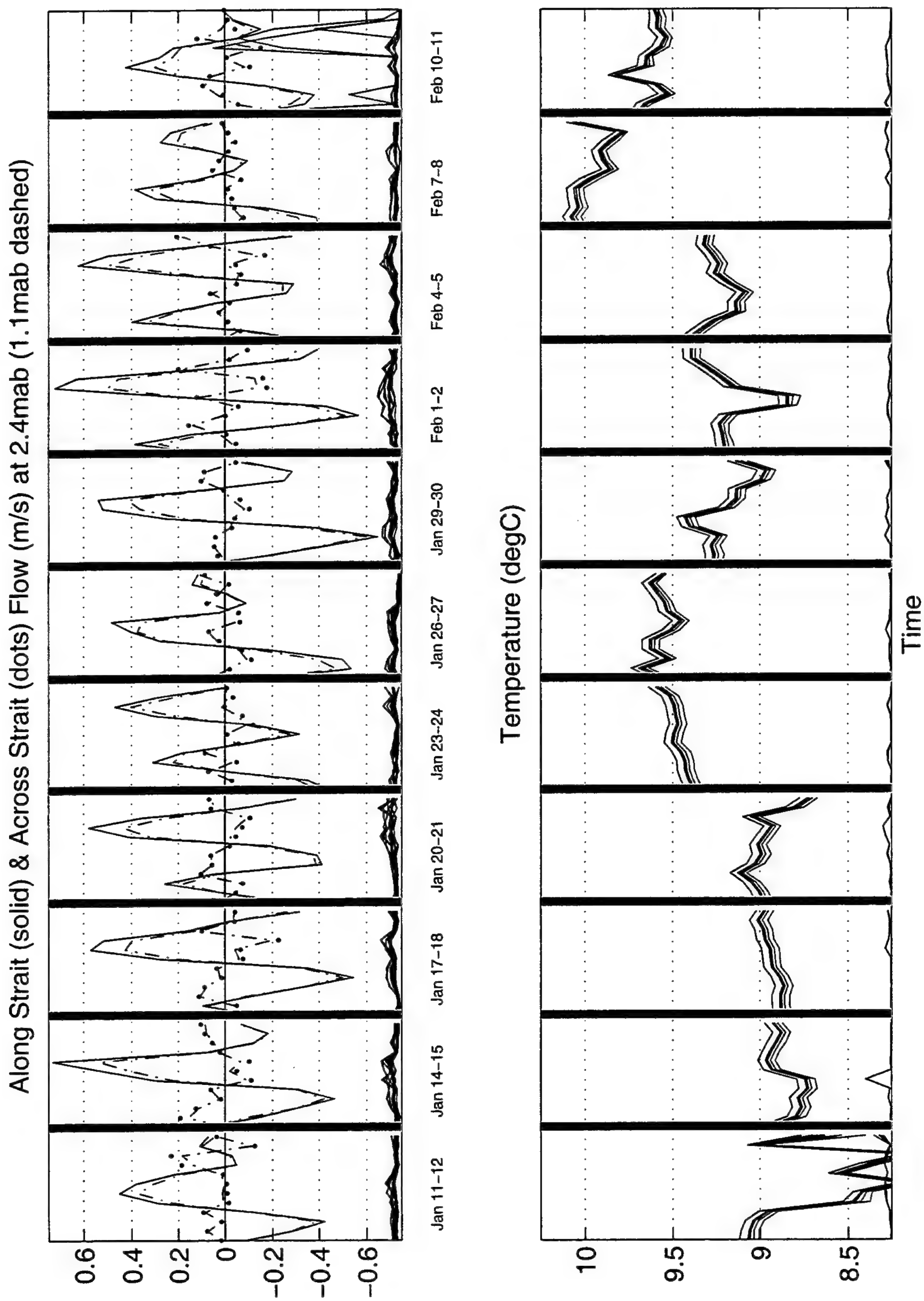
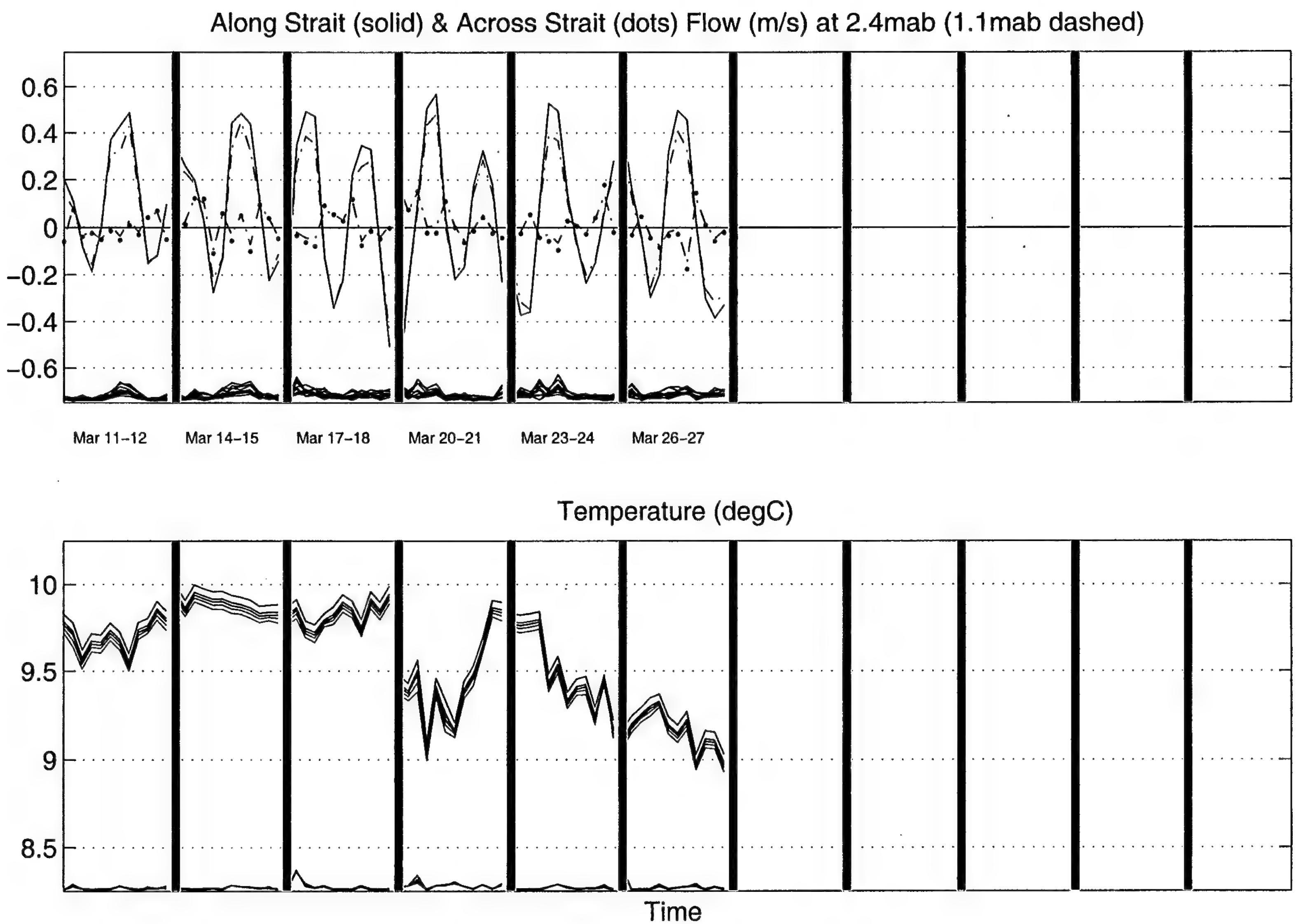

Along Strait (solid) & Across Strait (dots) Flow (m/s) at 2.4mab (1.1mab dashed)
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
Nov 11−12
Nov 14−15
Nov 17−18
Nov 20−21
Nov 23−24
Nov 26−27
Nov 29−30
Dec 2−3
Dec 5−6
Dec 8−9
Dec 11−12
Temperature (degC)
9
8.5
8
7.5
Time

Figure 17.



Along Strait (solid) & Across Strait (dots) Flow (m/s) at 2.4mab (1.1mab dashed)
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
Jan 11-12
Jan 14-15
Jan 17-18
Jan 20-21
Jan 23-24
Jan 26-27
Jan 29-30
Feb 1-2
Feb 4-5
Feb 7-8
Feb 10-11
Temperature (degC)
10
9.5
9
8.5
Time

Figure 18.



Along Strait (solid) & Across Strait (dots) Flow (m/s) at 2.4mab (1.1mab dashed)
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
Mar 11–12
Mar 14–15
Mar 17–18
Mar 20–21
Mar 23–24
Mar 26–27
Temperature (degC)
10
9.5
9
8.5
Time

Figure 19.

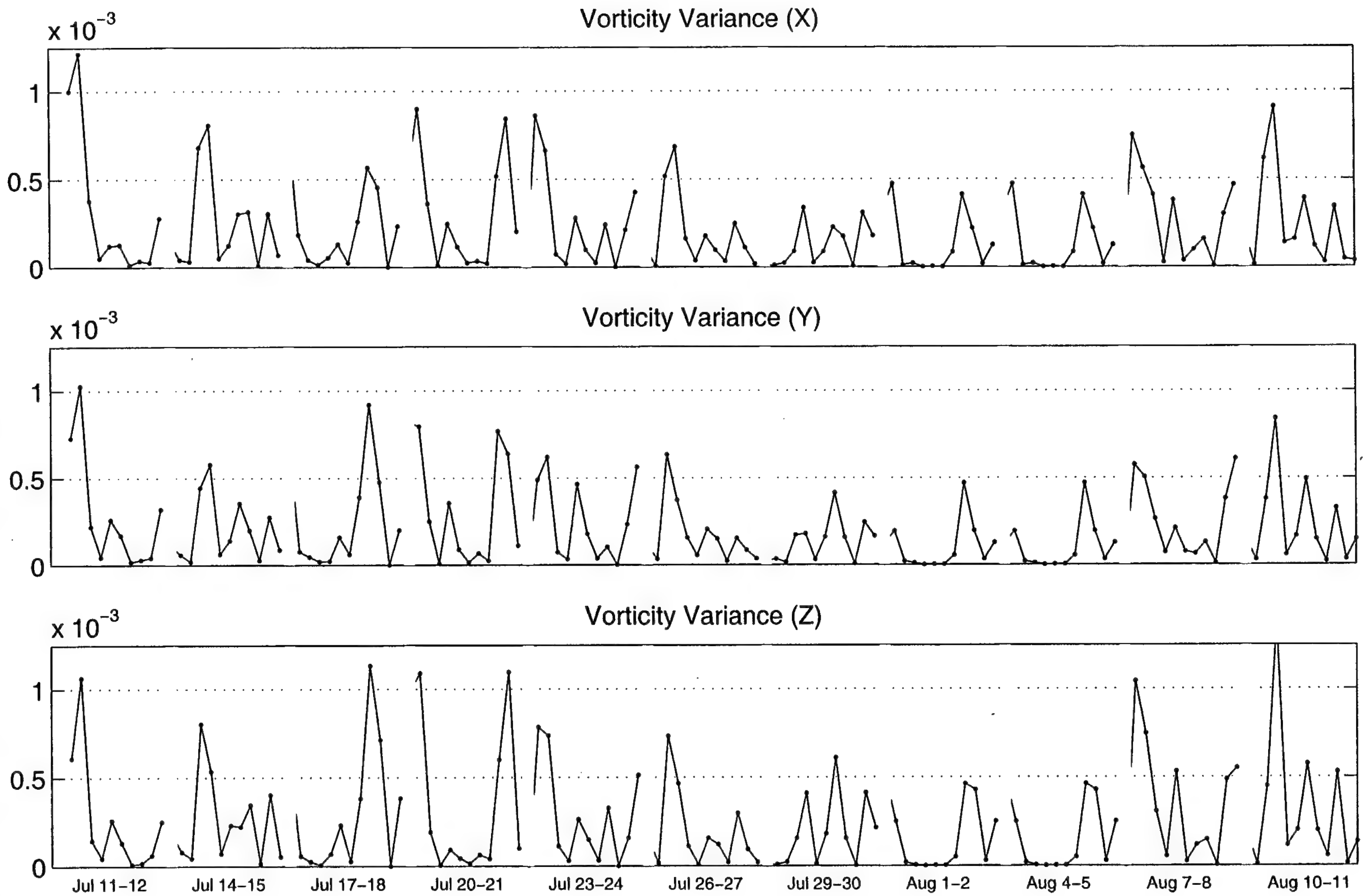
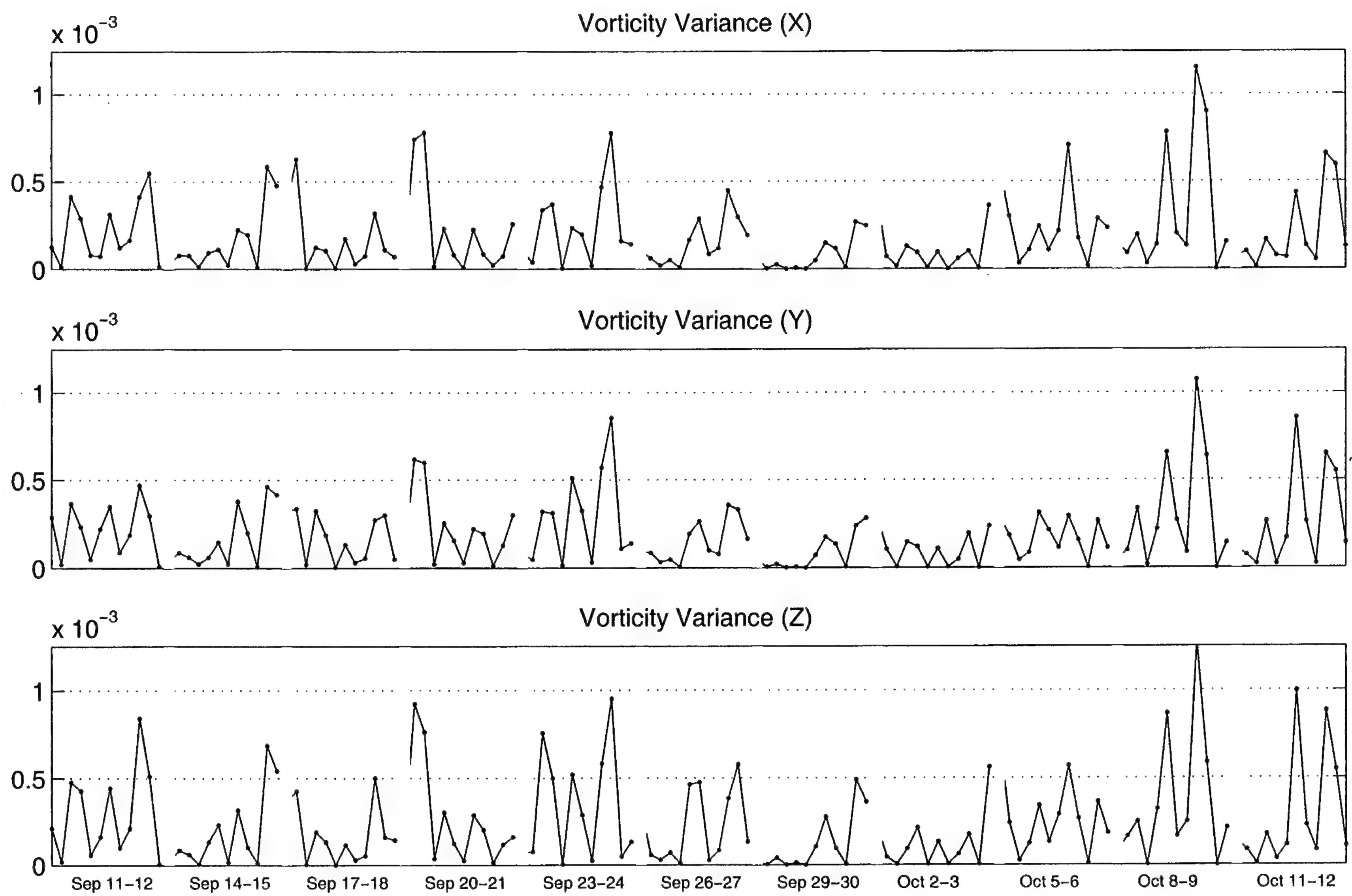

Vorticity Variance (X)
x 10^-3
1
0.5
0
Vorticity Variance (Y)
x 10^-3
1
0.5
0
Vorticity Variance (Z)
x 10^-3
1
0.5
0
Jul 11–12
Jul 14–15
Jul 17–18
Jul 20–21
Jul 23–24
Jul 26–27
Jul 29–30
Aug 1–2
Aug 4–5
Aug 7–8
Aug 10–11

Figure 20.



Vorticity Variance (X)
x 10^-3
1
0.5
0
Vorticity Variance (Y)
x 10^-3
1
0.5
0
Vorticity Variance (Z)
x 10^-3
1
0.5
0
Sep 11–12
Sep 14–15
Sep 17–18
Sep 20–21
Sep 23–24
Sep 26–27
Sep 29–30
Oct 2–3
Oct 5–6
Oct 8–9
Oct 11–12

Figure 21.

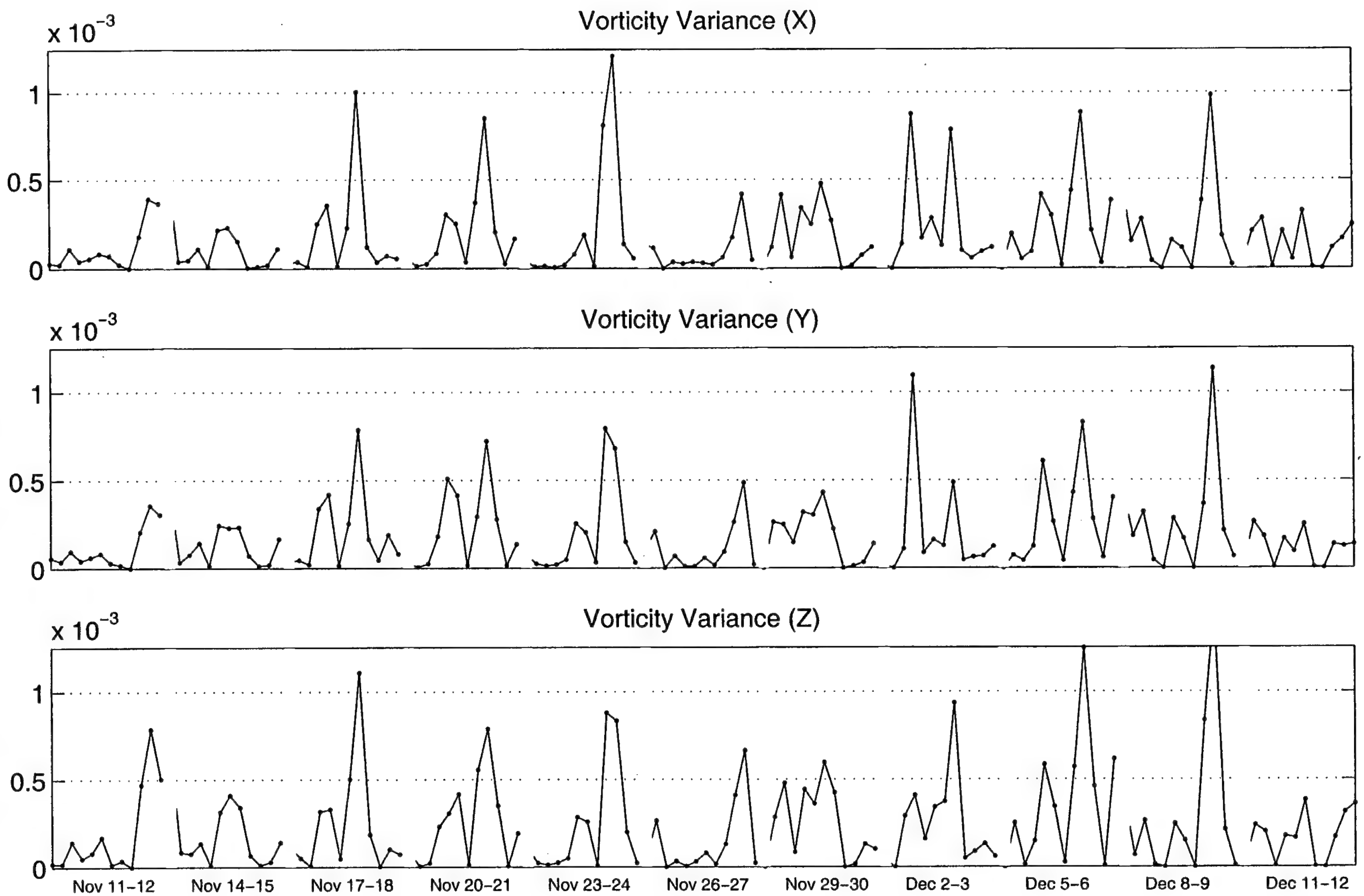

Vorticity Variance (X)
x 10^-3
1
0.5
0
Vorticity Variance (Y)
x 10^-3
1
0.5
0
Vorticity Variance (Z)
x 10^-3
1
0.5
0
Nov 11-12
Nov 14-15
Nov 17-18
Nov 20-21
Nov 23-24
Nov 26-27
Nov 29-30
Dec 2-3
Dec 5-6
Dec 8-9
Dec 11-12

Figure 22.

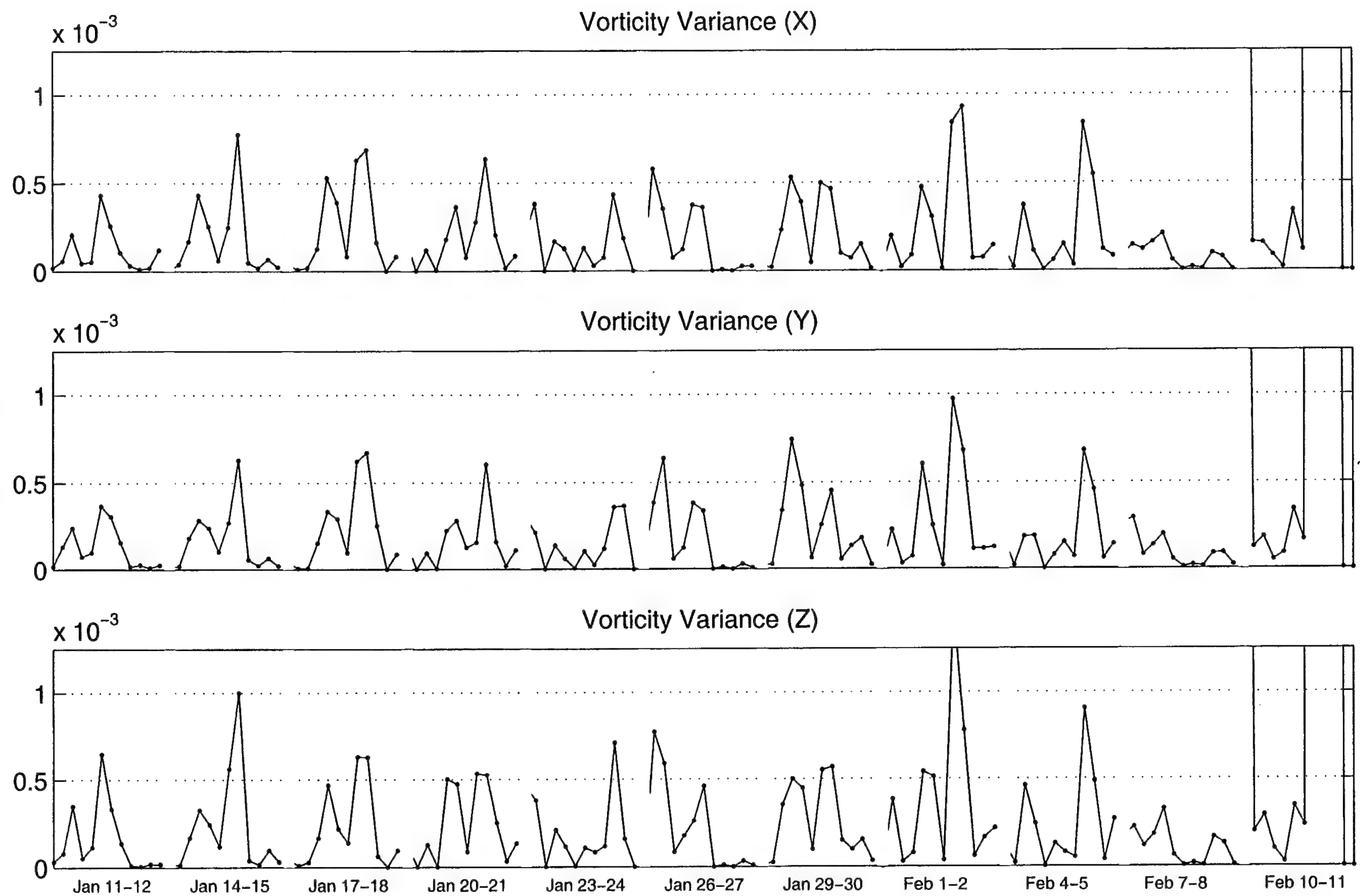
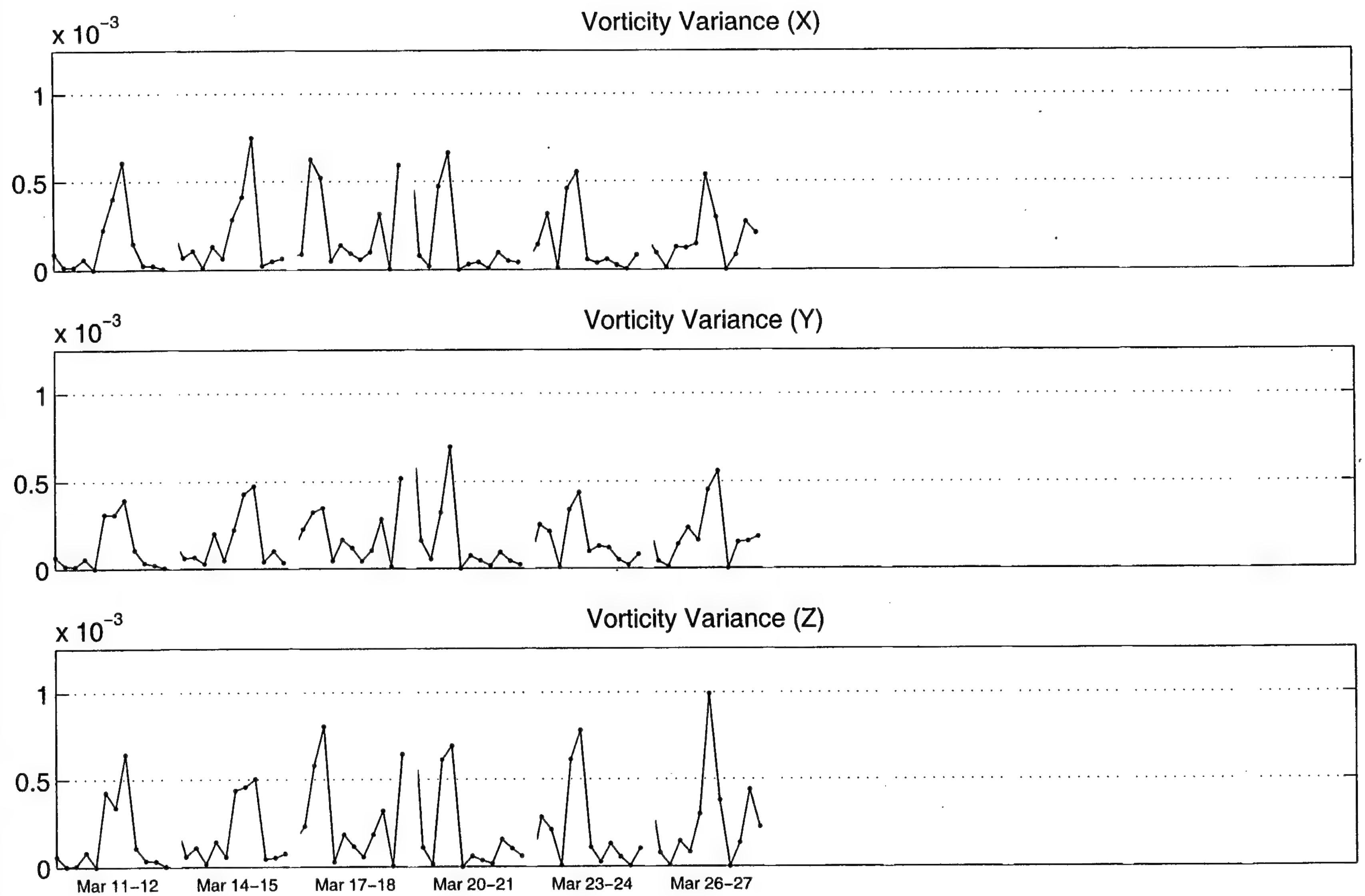

Vorticity Variance (X)
x 10^-3
1
0.5
0
Vorticity Variance (Y)
x 10^-3
1
0.5
0
Vorticity Variance (Z)
x 10^-3
1
0.5
0
Jan 11-12
Jan 14-15
Jan 17-18
Jan 20-21
Jan 23-24
Jan 26-27
Jan 29-30
Feb 1-2
Feb 4-5
Feb 7-8
Feb 10-11

Figure 23.



Vorticity Variance (X)
x 10^-3
1
0.5
0
Vorticity Variance (Y)
x 10^-3
1
0.5
0
Vorticity Variance (Z)
x 10^-3
1
0.5
0
Mar 11–12
Mar 14–15
Mar 17–18
Mar 20–21
Mar 23–24
Mar 26–27

Figure 24.

7. CONCLUSIONS

It is apparent that the spectral density of the along-path flow is filtered in a manner which is not yet well understood, albeit repeatable. Evaluation of vorticity spectra must remain restricted to regions below this filtering, until such time the filtering is understood. When flow is perpendicular to the acoustic path, vortex shedding appears to contaminate the spectra at wave numbers consistent with the tubing diameters of the tripod.

8. ACKNOWLEDGMENTS

Eugene Terray (WHOI) and Fredrik Thwaites (WHOI) are acknowledged for their major contributions to the theory and design of the vorticity meters.

We would like to thank the WHOI engineering and diving teams for their support: Allan Gordon, Don Peters, Fred Thwaites, Gary Stanbrough and Todd Morrison. We also appreciate the graphic illustrations done by Betsey Doherty (WHOI) and the thermistor calibration performed by Maren Tracy Plueddemann (WHOI CTD Lab).

For the voyage to the site and the logistical support , we thank the crew of the RV/Thompson of the University of Washington for deployment and to the staff and crew of the Far West Charter Company for the recovery of the pop-up floats.

This program was funded by the Marine Geology and Geophysics Program of the Office of Naval Research under ONR Grant: N00014-94-I-0436 to the Woods Hole Oceanographic Institution. This support is gratefully acknowledged.

9. REFERENCES

Batchelor, G.K. 1953. *The Theory of Homogeneous Turbulence*, Cambridge University Press, New York.

Bendat, Julius S. and Allan G. Piersol, 1986. *Random Data Analysis And Measurement Procedures (Second Edition)*, John Wiley & Sons, Inc., 566pp.

Kaimal, J.C., Wyngaard, J.C. & Haugen, D.A. 1968. "Deriving Power Spectra from a Three-component Sonic Anemometer", *Journal of Applied Meteorology*, Vol. **7**, pp.827-837.

Thwaites, Fredrik Turville, 1995. "Development of an Acoustic Vorticity Meter to Measure Shear in Ocean-Boundary Layers", Ph.D. Thesis. MIT/WHOI, 95-22, pp.187.

Williams, A.J. 3rd, John S. Tochko, Rickard L. Koehler, William D. Grant, Thomas F. Gross and Christopher V.R.Dunn. 1987. "Measurement of Turbulence in the Oceanic Bottom Boundary Layer with an Acoustic Current Meter Array", *Journal of Atmospheric and Oceanic Technology*, Vol. **4**. No. 2, 16pp.

APPENDIX A: Data Processing Software

```
% seepress - script to convert pressure counts to pressure (psia)
% for sparse sampling 17 sets * 514 samples /set
nn=1;

% dint when continuous sampling is made
dint = 514;


for fn=2:12
eval(['load v_doy_',num2str(fn)])
eval(['load v_press_',num2str(fn)])

% find break after 32 minutes of sampling
nbreaks = find(diff(doy) > 0.01);
nbreaks= [nbreaks;length(doy)]; .
nbegin=1;

for nendi=1:length(nbreaks) % for each 32 minute series
nend=nbreaks(nendi);

% if complete 32 minute cycle compute 3 avgs for each observation
for nstart=nbegin:dint:nend
   nstop=nstart+dint-1;
   if(nstop >= nend)
            nbegin = nbreaks(nendi)+1;
            break;
            end
   if (nstop > length(doy))
            nstop = length(doy);
            disp(['last sample only avgs ',num2str(nstop-nstart+1)])
           end
% converts 4Hz pressure counts to pressure (psia) for VORT#3
%
pc = pcounts(nstart:nstop);
p=diff(pc');
p=[p(1) p];
err=find(p < 0);
p(err)=p(err)+2^24;
frq = 1000 /140;
N=32 /frq;
T=(N*1e6). /p;
T2=T.^2;
T0=30.44344;
T02=T0^2;
C=-4858.177;
D=0.066768;
ddt=1-(T02. /T2);
press=C.*ddt.*(1-D.*ddt);
pstd(nn)=std(press);
pmean(nn)=mean(press);
mdoy(nn)=mean(doy(nstart:nstop));
nn=nn+1
end  % end of nstart loop (slip through hours dint at a time)
nbegin=nend+1;
if (nbegin >= max(nbreaks))
            break
            end
end % nbreak (nendi)

end  % end of each v_vel_N file
```

```
% gettemp – to convert thermistor counts from bin2mat to degrees C
%

load tempckV3

%   ck in tempckV3:
%ck =
%
%       0.0237    -0.2758     5.6379    -0.0182
%       0.0238    -0.2763     5.6439    -0.0453
%       0.0239    -0.2770     5.6352    -0.0009
%       0.0249    -0.2837     5.6509     0.0098
%       0.0251    -0.2852     5.6561    -0.0196
%       0.0250    -0.2846     5.6537    -0.0081
%       0.0252    -0.2863     5.6613    -0.0029
%       0.0247    -0.2822     5.6451    -0.0048

ck2=ck';
for fn=5:15
eval(['load v_doy_',num2str(fn)])
eval(['load v_temp_',num2str(fn)])
ntherms=length(temp) /length(doy)
%   for vort3v30 if (fn == 4) % problem at end of file
%   for vort3v30              ntherms=length(temp) /(length(doy)+1);
%   for vort3v30              end
nr = length(temp) /ntherms;
tempv=reshape(temp,ntherms,nr)'./10000;
clear temp
for ist=1:8
   temp(:,ist) = ck2(1,ist).*tempv(:,ist).^3+ck2(2,ist).*tempv(:,ist).^2+ck2(3,ist).*tempv(:,ist)+ck2(4,ist);
end

%if (fn == 4)
%           temp((length(doy)+1),:)=[];
%           end
eval(['save v_temp_',num2str(fn),' temp'])
end
```

```
% fixvort - to convert 0-2^16 to -1.5 -> +1.5 m /s .
%  to read output from bin2mat
% jfredericks@whoi.edu   7 /22 /94
nvels=length(velocity) /length(doy)
nr=length(velocity) /nvels
vel=reshape(velocity,nvels,nr)';
for ns=1:nvels
      nn=find(vel(:,ns) > 32768);
      if (length(nn))
         vel(nn,ns)=vel(nn,ns)-65536;
         end
      end

vel = vel.*(1.5 /32678);   % convert to vel (m /s)
```

getvelV3.m

```
% procedure to convert VORT#3 velocity data to VORT#2 orientation
%   upon completion real(w...r)  corresponds to X axis of VORT2
xtop=-vmean(:,3);
ytop=(vmean(:,11)-vmean(:,7)) /(2*cos(30*pi /180));
xtopcheck=(vmean(:,11)+vmean(:,7)). /(2*sin(30*pi /180));

xbot=vmean(:,1);
ybot=(vmean(:,5)-vmean(:,9)). /(2*cos(30*pi /180));
xbotcheck=-(vmean(:,5)+vmean(:,9)). /(2*sin(30*pi /180));

wtop=xtop+i*ytop;
wbot=xbot+i*ybot;

%degrees=45;  % for vort3v30.dat
%degrees=60;  % for vort3.617
%degrees=-56; % for vrt3pu.dat if using 3 /1 or 0 if using 9 /11
%rotated
%topr=wtop.*thetar;
%botr=wbot.*thetar;
```

APPENDIX B: Rotation of ABC vorticity planes to XYZ

Appendix B. Rotation of Vorticity Planes Into Along-Strait (x), Across-Strait(y) and Vertical (z) Coordinates

Assuming equidistant length between all sensor planes within each circulation path and each sensor within all horizontal planes, each sensor can be described as a point source as follows:

$$
\begin{aligned}
(X_b, Y_b, Z_b) &= (0,0,0)\\
(X_r, Y_r, Z_r) &= (L,0,0)\\
(X_o, Y_o, Z_o) &= \left(\frac{L}{2}, \frac{\sqrt{3}}{2}, 0\right)\\
(X_y, Y_y, Z_y) &= \left(0, \left(\frac{1}{\sqrt{3}}\right)L, \frac{\sqrt{2}}{\sqrt{3}}L\right)\\
(X_w, Y_w, Z_w) &= \left(L, \frac{1}{\sqrt{3}}L, \frac{\sqrt{2}}{\sqrt{3}}L\right)
\end{aligned}
$$

Each vorticity vector can be defined as follows:

$$
\begin{aligned}
\vec{a} &= \text{vector of unit length in A direction} = (a_x, a_y, a_z)\\
\vec{b} &= \text{vector of unit length in B direction} = (b_x, b_y, b_z)\\
\vec{c} &= \text{vector of unit length in C direction} = (c_x, c_y, c_z)
\end{aligned}
$$

It can be shown that

$$
R = \begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{\sqrt{2}}{2} \\ 0 & -1 & \frac{\sqrt{2}}{2} \end{bmatrix} \bullet \sqrt{\frac{2}{3}}
$$

and

$$
\omega_{xyz} = R^{-1} \bullet \omega_{abc}
$$

APPENDIX C: Data Logging & Unpacking Software

```
/* bin2mat - read VORT#3 or VORT#2 binary data files
             & output MATLAB files of time /pressure /vel /temp */

/* jjfredericks Woods Hole Oceanographic Institution
        jfredericks@whoi.edu                6 /94 */
/* NOTE:  must recompile with FLAGB set for VORT3 or VORT2 */

#include <stdio.h>
#include <strings.h>
#include <malloc.h>
#include "mat.h"  /* symbolic link to /usr /local /matlab /extern /...*/

#define FILESIZE 229376  /* No. bytes per disk dump*/
#define MAXFILES 1481    /* maximum number of disk dumps */
#define ENDBUFF 256      /* will not begin a data record beyond FILESIZE - ENDBUFF */

#define MAXREC 130000      /* 514*17*12=104856 added extra */
#define RECLENV3 58  /* Vort#3 */
#define RECLENV2 74  /* Vort#2 */


#define FLAGA 0xab
#define FLAGB 0x53
/* #define FLAGB 0x52 for VORT#2 */

#define NTIMEWORD 6        /* number of bytes in time word */

/* monthly table for computation of day_of_year */
unsigned short monthday[]={0,31,60,91,121,152,182,213,244,274,305,335};

unsigned char time[NTIMEWORD],oldsec;
unsigned long nrec;
int nvels,ntherms,namex;
double doy[MAXREC],*velptr,*temptr,*vels,*temps,pitch[MAXREC], roll[MAXREC],
        compass[MAXREC], pcounts[MAXREC];
float fsec,stoptime,startime,atof(),thisdoy,lastdoy;

MATFile *fp;
Matrix *a,*b;

char file_out[40];
```

main

```
main(argc,argv)
int argc;
char **argv;
{
    unsigned char dread[FILESIZE];
    int nb,ic,dsize,nc,status;

/* fsec is for the addition of the 140 ms sampling rate between each second */


    fprintf(stderr,
        "\n\n\n TATTLETALE BINARY TO MATLAB BINARY CONVERSION   9/1/94\n\n");

    startime=0;
    stoptime=500.0;
        fprintf(stderr,"sizeof double: %ld\n",sizeof(double));

    if (argc < 2)
        { fprintf(stderr,
```

```
        "\nUSAGE:   bin2mat {v2|v3} [startime stoptime] <packed_datafile\n\n");
        exit(1);
        }


    if (strcmp(*++argv,"v2")==0)
        {
          fprintf(stderr,
              "Reading data in VORT#2 format/24 acoustic paths/4 thermistors\n");
        dsize=RECLENV2;
        ntherms=4;
        nvels=24;
        }
    else if (strcmp(*argv,"v3")==0)
        {
        dsize=RECLENV3;
        ntherms=8;
        nvels=12;
          fprintf(stderr,
 "Reading (%d byte) datablocks in VORT#3 format/%d acoustic paths/%d thermistors\n",
                    dsize,nvels,ntherms);
        }
    else
          { fprintf(stderr,
          "\nUSAGE:   bin2mat {v2|v3} [startime(doy) stoptime(doy)] <packed_datafile\n\n");
          exit(1);
          }
    if (argc > 2)
          {
          startime=atof(*++argv);
          fprintf(stderr," Start time specified: %g\n",startime);
          if (argc > 3)
            {
            stoptime= atof(*++argv);
            fprintf(stderr," Stop time specified: %g\n",stoptime);
            }
          }

    lastdoy=startime;

     /* allocate space for temp & vel variable ncols * /
     vels = (double *) calloc(nvels*MAXREC,sizeof(double));
     temps= (double *) calloc(ntherms*MAXREC,sizeof(double));
        velptr = vels;
        temptr = temps;
          fprintf(stderr,"vels: %d/ velptr: %d\n",vels,velptr);
          fprintf(stderr,"temps: %d/ temptr: %d\n",temps,temptr);
     if (vels == 0 || temps == 0)
          {
          fprintf(stderr,"trouble allocating memory at rec: %d \n",nrec);
          exit(1);
          }

nrec = 0;
namex = 0;
oldsec= (unsigned char) 0;

for (nb=0; nb < MAXFILES; nb++)
{
          /* read in a disk block * /
          status = fread(dread,sizeof(char), FILESIZE, stdin );
          if (status <= 0)
                    {   fprintf(stderr,
          "Error searching for rec %d nblock = %d status = %d\n",
```

```
                nrec,nb,status);
                }

        /* find beginning of valid records */
    for (ic=0; ic < FILESIZE; ic++)
        {
        if (((dread[ic] == FLAGA) && (dread[ic+1] == FLAGB)) &&
                ((dread[ic+dsize] == FLAGA) && (dread[ic+dsize+1] == FLAGB)))
                break;
        }


    if (ic > FILESIZE - ENDBUFF)    /* no records in block */
                {
                fprintf(stderr," LOST IN FILE   at block: %d\n",nb);
                nrec-=1;
                dodump();
                exit(0);
                }

    if (status >= dsize)
        fprintf(stderr,
                "found first byte at %d in block %d\n",ic,nb);

    ic++;   /* found valid data now skip over flag 0xab */


/*  BEGIN getting RECORDS from BLOCK */

    while (ic < FILESIZE-ENDBUFF)
    {
    if(status >= dsize)
      {
        /* read time: MO DA HR MN SEC ff */
    for (nc=0; nc < NTIMEWORD; nc++)
            time[nc]=dread[++ic];
    if(time[4] == oldsec)   /* fix time for 140 ms sampling rate */
            fsec+=0.140;
    else
        {
        fsec=0.0;
        oldsec=time[4];
        }
    thisdoy=doy[nrec]=(double) (monthday[time[0]-1]+time[1]+time[2] /24.0+
                time[3] /1440.0+((double) time[4]+fsec) /86400.0);


      ic++;   /* advance pointer to read next byte */
    for (nc=0; nc < ntherms; nc++)
         {
         if(thisdoy >= startime)
           *temptr++=(double) (dread[ic++]+dread[ic++]*256);
         else
                {ic++;ic++;}
           }

    pitch[nrec]=( (double) (dread[ic++]+dread[ic++]*256)-49152) /16384*45;
    roll[nrec]=( (double) (dread[ic++]+dread[ic++]*256-49152) /16384*45);

    for (nc=0; nc < nvels; nc++)
           {
           if (thisdoy >= startime)
              *velptr++= (double) (dread[ic++]+dread[ic++]*256);
```

```
            else
                {ic++;ic++;}
            }
    compass[nrec]= (double) (dread[ic++]+dread[ic++]*256);
      pcounts[nrec]=(double) (dread[ic++]*65536+dread[ic++]*16777216+dread[ic++]+dread[ic++]*256);

/*    force dump of stored records if
                reached stoptime ... or ...
                reached a 1.5 day gap ... or ...
                read the last file block    */

    if(lastdoy >= stoptime)
        {
        dodump();
        fprintf(stderr,"gone beyond stoptime\n");
        exit(1);
        }

    if((thisdoy-lastdoy) > 1.5 && thisdoy > startime)
                dodump();  /* a gap of > 1.5 days */

    if(lastdoy >= startime)
                nrec++;

    if(nrec >= MAXREC && thisdoy > startime)
                {
                dodump();
                exit(0);
                }
    }   /* end of if (status >= dsize)   */

    if(status != FILESIZE)
       /* ran out of file */
                {
                if(nrec) dodump();
                fprintf(stderr,
                   "reached partial record so quit reading\n");
                exit(0);
                }


/*
          fprintf(stderr,"[dread[%d]: %0x ... %0x\n",ic,dread[ic],dread[ic+1]);
*/
     lastdoy=thisdoy;

    if(ic > 355 && ic < 365)    /* hack a discovered problem   */
          {
          fprintf(stderr,"hacked code at nb:%d \nrec: %d\n",
                nb,nrec);
         ic = ic-2;
          }

     if(dread[ic++] != FLAGA && dread[ic] != FLAGB)
                          break;   /* check for end of data in block */
     };   /* finished with data part of disk block */

  }   /* finished with disk block */

fprintf(stderr,"NBLOCKS READ: %d, last record contains %d bytes\n",nb,nrec);
```

```
}       /* End of Main Loop */


/* routine to dump to datafiles */

dodump()                                                    dodump
                    {
                    int ic;
                     if((thisdoy-lastdoy) <= 1.5)    /* not at multiday gap */
                          nrec += 1;  /* convert index to number of records */
                               /* don't need to if new day because we don't
                                    want to dump the last record stored */
                      printf("month(%d)/day(%d)/hr(%d)/min(%d)/sec(%d) (%d records)\n",
                                time[0],time[1],time[2],time[3],time[4],nrec);

                    a=mxCreateFull(nrec,1,REAL);
                    sprintf(file_out,"%c_doy_%d.mat",'v',namex);
                    fp=matOpen(file_out,"w");
                    memcpy(mxGetPr(a),doy,nrec*sizeof(double));
                    mxSetName(a,"doy");
                    matPutMatrix(fp,a);
                    matClose(fp);
                     mxFreeMatrix(a);

                    sprintf(file_out,"%c_vel_%d.mat",'v',namex);
                    fp=matOpen(file_out,"w");
                    a=mxCreateFull((nrec)*nvels,1,REAL);
                    mxSetName(a,"velocity");
                    memcpy(mxGetPr(a),vels,nrec*nvels*sizeof(double));
                    matPutMatrix(fp,a);
                    matClose(fp);
                     mxFreeMatrix(a);

                    sprintf(file_out,"%c_temp_%d.mat",'v',namex);
                    fp=matOpen(file_out,"w");
                    a=mxCreateFull((nrec)*ntherms,1,REAL);
                    mxSetName(a,"temp");
                    memcpy(mxGetPr(a),temps,nrec*ntherms*sizeof(double));
                    matPutMatrix(fp,a);
                    matClose(fp);
                     mxFreeMatrix(a);

                    sprintf(file_out,"%c_press_%d.mat",'v',namex);
                    fp=matOpen(file_out,"w");
                    a=mxCreateFull(nrec,1,REAL);
                    mxSetName(a,"pcounts");
                    memcpy(mxGetPr(a),pcounts,nrec*sizeof(double));
                    matPutMatrix(fp,a);
                    matClose(fp);
                     mxFreeMatrix(a);

                    sprintf(file_out,"%c_prc_%d.mat",'v',namex);
                    fp=matOpen(file_out,"w");
                    a=mxCreateFull(nrec,1,REAL);
                    mxSetName(a,"pitch");
                    memcpy(mxGetPr(a),pitch,nrec*sizeof(double));
                    matPutMatrix(fp,a);
                    mxSetName(a,"roll");
                    memcpy(mxGetPr(a),roll,nrec*sizeof(double));
                    matPutMatrix(fp,a);
                    mxSetName(a,"compass");
                    memcpy(mxGetPr(a),compass,nrec*sizeof(double));
```

```
    matPutMatrix(fp,a);
    matClose(fp);
     mxFreeMatrix(a);

    namex++;
if((thisdoy-lastdoy) <= 1.5)    /* not a inter day gap */
       {
    fprintf(stderr,"nrec: %d\nvelptr: %ld\ntemptr: %ld\n",
                nrec,velptr,temptr);
        nrec=-1;
        velptr = vels;
        temptr = temps;
       }
 else   /* copy last to first for next day */
       {
        doy[0]=doy[nrec];
        pitch[0]=pitch[nrec];
        roll[0]=roll[nrec];
        compass[0]=compass[nrec];
        pcounts[0]=pcounts[nrec];

        for (ic=0; ic < nvels; ic++)
             velptr--;
        for (ic=0; ic < ntherms; ic++)
                temptr--;

        memcpy(vels,velptr,nvels*sizeof(double));
        memcpy(temps,temptr,ntherms*sizeof(double));

    fprintf(stderr,"nrec: %d\nvelptr: %ld\ntemptr: %ld\n",
                nrec,velptr,temptr);

        nrec=0;
        velptr = vels;
        temptr = temps;
        for (ic=0; ic < nvels; ic++)
                velptr++;
        for (ic=0; ic < ntherms; ic++)
                temptr++;
    fprintf(stderr,"nrec: %d\nvelptr: %ld\ntemptr: %ld\n",
                nrec,velptr,temptr);
       }
    }   /* end of dump to matlab files */
```

```
 10 PRINT " VORT3 BASS, 26-JUNE-94 ROM VERSION from IMCSBASS 17-March-94"
 11 PRINT " 514 RECORDS EVERY 2 MIN FOR 16 BLOCKS (32 MIN) EVERY 2 HR,"
 12 PRINT " EVERY 3 DAYS, EVERY OTHER MONTH."
'20 Q=&H4000:REM DATAFILE STARTS AT 4000 IN RAM
 20 Q=&H0143:REM DATAFILE STARTS AT 0143 IN ROM

'F=PWR FLAG, 0 FOR NOTHING ON, 1 FOR P&S, 2 FOR P,S,&O.
'D=DO CALIBRATE A/D AND READ COMPASS LATCH
'ACMs,OBSs,PRES,& SEABIRDs MEASURED FOR 15 MINUTES EVERY HOUR AT 4 Hz.
'TO LATCH PWR, FIRST PWR ON AND WASTE TIME WITH:
'PSET 16:CALL &HFFD9,0:CALL &HFFD9,0, THEN:
'COMPASS ON      PSET 15
'PRESS ON        PSET 7,14:PCLR 7,16,14
'SEABIRDS ON     PSET 6,14:PCLR 6,16,14
'OBSs ON         PSET 2,14,:PCLR 2,16,14.
'TO PREVENT TURNING THESE OFF WHEN SELECTING AUX WITH D\6,
'HOLD D\4 HI WITH PSET 16:CALL &HFFD9,0:CALL &HFFD9,0
'PSET 1,14:PCLR 1,16 WHERE CALL &HFFD9 IS A 2.2MS TEST OF UART BUFFER.
100 ASM &H9E,DB &H00:REM CONTROL DISABLE
102 SLEEP 0:PCLR 0,1,2,3,4,5,6,7,8,14,15:PSET 14,16:REM ESTABLISH DDRs
108 SLEEP 240:PCLR 14,16:REM LET A/D CALIBRATE FINISH IF STARTED.
110 PRINT "   TYPE   WAKE   TO GET CONTROL (5sec)"
112 X = 0: STORE X,#4,&H01010101 : X = 0: ITEXT X,500
114 X = 0: IF GET(X,#4)<>&H57414B45 GOTO 300

130 RTIME
131 PRINT #02,"OLD TIME=",?(3),"/",?(4),"/",?(5)," ",?(2),":",?(1),":",?(0)
132 INPUT " SET TIME? 1=YES",J
133 IF J<> 1 GOTO 160
134 INPUT " THE YEAR 0-99    ",?(5)
136 INPUT " THE MONTH 1-12  ",?(4)
138 INPUT " THE DAY 1-31  ",?(3)
140 INPUT " THE HOUR 0-23    ",?(2)
142 INPUT " THE MINUTE 0-59 ",?(1)
144 INPUT " THE SECOND 0-59 ",?(0)
150 STIME
160 RTIME
162 PRINT #02,"NEW TIME=",?(3),"/",?(4),"/",?(5)," ",?(2),":",?(1),":",?(0)
164 INPUT " STOP? 1=YES",J
166 IF J<> 1 GOTO 300
230 STOP

300 ONERR 100
1000 DIM(1024,8192)
' ASSEMBLY ROUTINES
1010 X=0:A=0:B=0:C=0:D=0:E=514:F=0:G=0:H=0:I=0:J=0:K=0:L=0:M=0:N=0:O=0
1015 PRINT "    A    B    C    D    E    F    G    H";
1016 PRINT "    I    J    K    L    M    N    O    P    T    X"
1020 GOSUB 8000:REM FIRST PASS
1025 PRINT #5H,A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,T,X
1030 GOSUB 8000:REM SECOND PASS
1031 PRINT #5H,A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,T,X
1050 N=0:REM INITIALIZE COUNTER
1080 GOTO 2000

'SUBROUTINE TO TURN ON PRESSURE AND SEABIRDS
1200 PSET 16:CALL &HFFD9,0:CALL &HFFD9,0:REM LET PWR SETTLE
1210 PSET 6,7,14:PCLR 6,7,16,14:F=1:RETURN
```

```
'SUBROUTINE TO TURN ON OBSs, PRESSURE, AND SEABIRDS
1230 PSET 16:CALL &HFFD9,0:CALL &HFFD9,0:REM LET PWR SETTLE
1240 PSET 2,6,7,14:PCLR 2,6,7,16,14:F=2:RETURN

'SUBROUTINE TO TURN OFF EVERYTHING
1250 PCLR 2,6,7,14:PSET 14:PCLR 14:F=0:RETURN

'SUBROUTINE TO ENABLE AUXILLIARIES ON MULTIPLEXOR
1260 PSET 16:REM LET PWR SETTLE
1270 PSET 1,14:PCLR 1,16:RETURN:REM DISABLES POWER LATCH

' CALIBRATE A/D
2000 SLEEP 0:PSET 15:SLEEP 600:PSET 16:REM PWR ON,CLEAR LATCHED PWR
2050 CALL T,0:REM FIND COMPASS GAP
2060 Z=COUNT(40)-1:SLEEP 50:REM Z=aZimuth,COMPASS
'2065 PRINT "GOT COMPASS GAP"
2070 PCLR 15:SLEEP 240:REM A/D CALIBRATE;2,882,040 CYCLES AT 1.2288 MHz
2080 PCLR 14,16:REM POWER OFF AND PREPARE TO LATCH PWR
2082 D=0:REM CLEAR LATCH TO DO A/D AND Zn
2090 GOSUB 1200:REM TURN ON SEABIRD,PRESS
2100 SLEEP 0:P=SDI(24):REM SAMPLE AND READ PRESSURE

3000 RTIME:IF E>0 GOTO 3040
3001 IF ?(4)%2=0 GOTO 3030:REM RUN IN JA,MAR,MAY,JUL,S (JUL,S,N,JA,MAR)
3006 IF ?(3)%3<>1 GOTO 3030:REM RUN EVERY THIRD DAY
3008 IF ?(2)%2>0 GOTO 3030:REM ONLY RUN EVERY OTHER HOUR
3010 IF ?(1)>32 D=1:GOTO 3030:REM OFF MINUTE 33 TO END OF HOUR
3012 IF ?(0)>0 GOTO 3030:REM CONTINUE
3014 IF ?(1)%2=0 E=514
3015 IF D=1 GOTO 2000:REM COMPASS AND A/D ONCE/HR
3020 GOTO 2090:REM POWER ON PRESSURE AND START CYCLE

3030 GOSUB 1250:SLEEP 100:GOTO 3000:REM EVERYTHING OFF

3040 E=E-1:X=0:REM INITIALIZE DATAFILE
3050 STORE X,#2,&HAB53:REM KEY WORD AB V3 FOR VORT3 OR AB53
3051 STORE X,#1,?(4):REM MONTH
3052 STORE X,#1,?(3):REM DAYS
3055 STORE X,#1,?(2):REM HOURS
3060 STORE X,#1,?(1):REM MINUTES
3070 STORE X,#1,?(0):REM SECONDS
3080 STORE X,#1,N:REM COUNTER

3100 C=&H73000000+Q+X:REM MUX ADDRESS AND DATAFILE POINTER
3110 CALL A,C,X:REM CALL A/D ROUTINE,ACM THEN AUX,POWER OFF AT END
3111 X=X%&H10000-Q:REM GET BACK THE DATAFILE POINTER
' SUBTRACT ROUTINE (CALL B) EXPECTS FLAGGED DATA AT 6F38.
3119 C=&H73000000+Q+X:REM DATAFILE LOCATION
3120 CALL B,C,X:REM CALL SUBTRACT AND TRANSFER
3121 X=X%&H10000-Q:REM GET BACK THE BASIC DATAFILE POINTER
3125 STORE X,#2,Z:REM COMPASS
3130 STORE X,#4,P:REM PRESSURE

'3170 X=0:REM REINITIALIZE DATAFILE
'3180 PRINT #4H,GET(X,#2):REM KEYWORD
'3185 PRINT #02,GET(X,#1)," ",GET(X,#1)," ";:REM MONTH DAY
'3190 PRINT #02,GET(X,#1),":",GET(X,#1),":",GET(X,#1)," ";:REM HR:MM:SS
'3200 PRINT #02H, GET(X,#1):REM COUNT
'3205 FOR L=1 TO 10:REM T(0:7),PITCH,ROLL
```

```
'3206 PRINT #04H,GET(X,#2)," ";
'3207 NEXT L:PRINT
'3210 FOR M=1 TO 3:REM 1 VORT SENSOR
'3220 FOR L=1 TO 4:REM 4 WORDS/LINE
'3230 PRINT #04H,GET(X,#2)," ";:REM EACH SENSOR ON A LINE
'3240 NEXT L:PRINT:REM FOUR AXES/LINE
'3250 NEXT M
'3262 PRINT #3,GET(X,#2)," ";:REM COMPASS
'3264 PRINT #08H,GET(X,#4)," ";:REM PRESSURE
'3275 PRINT:REM SPACE
3280 PRINT {0,57};:REM TAKES 59 ms.

3290 N=N+1
3295 SLEEP 14:P=SDI(24):REM SAMPLE COUNTERS AND READ PRESSURE
3300 GOTO 3000

8000 X=&H7300:REM MULTIPLEXOR LIST
8010 ASM X,DW &H0000;DW &H0101;DW &H0202;DW &H0303:REM ACM 1/T1-T4
8015 ASM X,DW &H0808;DW &H0909;DW &H0A0A;DW &H0B0B:REM ACM 1/T5-T8
8020 ASM X,DW &H208A;DW &H218B;DW &H22FF;DW &H23FF:REM ACM 2/PITCH, ROLL
8025 ASM X,DW &H28FF;DW &H29FF;DW &H2AFF;DW &H2BFF
8030 ASM X,DW &H4000;DW &H4100;DW &H4200;DW &H4300:REM ACM 3
8035 ASM X,DW &H4800;DW &H4900;DW &H4A00;DW &H4B00
'8040 ASM X,DW &H6000;DW &H6100;DW &H6200;DW &H6300:REM ACM 4
'8045 ASM X,DW &H6800;DW &H6900;DW &H6A00;DW &H6B00
'8050 ASM X,DW &H8000;DW &H8100;DW &H8200;DW &H8300:REM ACM 5
'8060 ASM X,DW &H8800;DW &H8900;DW &H8A00;DW &H8B00
'8070 ASM X,DW &HA000;DW &HA100;DW &HA200;DW &HA300:REM ACM 6
'E8080 ASM X,DW &HA800;DW &HA900;DW &HAA00;DW &HAB00
8090 ASM X,DW &HFF00;DW &HFF00:REM END OF LIST

8100 X=&H3800:A=X:REM A/D ROUTINE
'   X IS POINTER TO AUX ARRAY, AB IS POINTER TO FLAGGED CONVERSION STORAGE
8105 ASM X,STD &H7000:REM POINTER TO DATAFILE
8107 ASM X,LDD #&H6F38;STD &H7002:REM POINTER TO FLAGGED CONVERSIONS
8110 ASM X,OIM &H04,&H17;PSHX;LDX #&H0600:REM POWER ON, WAIT 5ms
8120 D=X:ASM X,DEX;BNE D;PULX:REM WAIT (4*.81us/LOOP)
8150 ASM X,CLR &H11;LDAA #&H10;STAA &H10:REM DISABLE UART, SET SCI FOR
8160 ASM X,OIM 8,&H11:REM INTERNALLY CLOCKED 8 BIT DATA RECEIVE ENABLE
'DUMMY PULSE SENT OUT TO INITIALIZE
8170 ASM X,OIM &HAB,&H15;AIM &HBF,&H15:REM DUMMY AUX LOCATION
8171 ASM X,OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03
'START TIMING P27=1
8172 ASM X,AIM &H7F,&H03:REM REMOVE PULSE P27=0
8174 D=X:ASM X,LDAB #&H02;BITB &H03;BEQ D:REM CHECK P21 FOR DONE
8179 C=X:ASM X,AIM &H14,&H15:REM ENTRY POINT FOR AUX,MASK PORT 5
8180 ASM X,LDAA &H00,X;INX;INX:REM TOP OF MUX LIST,MOVE TO NEXT
8185 ASM X,ORAA &H15;STAA &H15:REM PUT MUX ON PORT 5
8200 ASM X,OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03
'START TIMING P27=1
8210 ASM X,AIM &H7F,&H03:REM REMOVE PULSE P27=0

8220 D=X:ASM X,LDAB #&H02;BITB &H03;BEQ D:REM CHECK P21 FOR DONE

8240 ASM X,LDAB &H15;ANDB #&H14:REM MASKED PORT 5 IN B
8250 ASM X,LDAA &H00,X;INX;INX:REM GET NEXT MUX WORD,ADVANCE
8255 ASM X,ABA;STAA &H15:REM ADD MUX WORD TO MASKED PORT 5, STORE P5
8280 ASM X,OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03
'
```

```
START TIMING P27=1
8290 ASM X,AIM &H7F,&H03:REM REMOVE PULSE P27=0

8400 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8402 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8403 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8404 ASM X,CPX 0;CPX 0;CPX 0;CPX 0

8410 ASM X,LDAA &H11;LDAB &H12:REM TRIGGER A READ
8420 ASM X,CPX 0;CPX 0;CPX 0;CPX 0:REM 16ECLOCK
8430 ASM X,LDAA &H11;LDAB &H12:REM HIGH BYTE IN B AND READ NEXT
8440 ASM X,CPX 0;CPX 0;CPX 0;CPX 0:REM 16ECLOCK
8450 ASM X,LDAA &H12:REM LOW BYTE IN A  DON'T READ NEXT
'REM FLIP THE BITS OF THE WORDS
8451 ASM X,ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA;RORB
8452 ASM X,ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA

8480 ASM X,TIM &H20,&H03;BEQ G:REM TEST BOTH RECEIVED
8488 ASM X,XGDX;RORB;BCS M:REM BRANCH IF AUXILLIARY (ODD X)
8490 ASM X,ROLB;XGDX;CLRA;CLRB;BRA G:REM IF NOT RECEIVED,CLEAR
8495 M=X:ASM X,ROLB;XGDX
' PUT AWAY TO @(50) ARRAY  THE FLAGGED VALUES
8510 G=X:ASM X,PSHX;LDX &H7002;STD 0,X;INX;INX;STX &H7002;PULX

8520 ASM X,LDAB #&HFF;EORB &H00,X;BEQ L;JMP D:REM CHECK FOR END OF LIST

8550 L=X:ASM X,LDAB #&H02:REM TEST FOR HOLD PULSE
8555 D=X:ASM X,BITB &H03;BEQ D:REM CHECK P21

8572 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8573 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8574 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8575 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8580 ASM X,LDAA &H11; LDAB &H12:REM TRIGGER A READ
8590 ASM X,CPX 0;CPX 0;CPX 0;CPX 0:REM 16ECLOCK
8600 ASM X,LDAA &H11;LDAB &H12:REM HIGH BYTE IN B AND READ NEXT
8610 ASM X,CPX 0;CPX 0;CPX 0;CPX 0:REM 16ECLOCK
8620 ASM X,LDAA &H12:REM LOW BYTE IN A DON'T READ NEXT
'REM FLIP THE BITS OF THE WORDS PUTS HIGH BYTE IN A, LOW BYTE IN B
8621 ASM X,ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA;RORB
8622 ASM X,ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA

8630 ASM X,TIM &H20,&H03;BEQ H:REM TEST BOTH RECEIVED
8640 ASM X,XGDX;RORB;BCS N:REM BRANCH IF AUXILLIARY (ODD X)
8645 ASM X,ROLB;XGDX;CLRA;CLRB;BRA H:REM IF NOT RECEIVED,CLEAR
8650 N=X:ASM X,ROLB;XGDX

' FINISH PUT AWAY TO @(50) ARRAY  THE FLAGGED VALUES
8660 H=X:ASM X,PSHX;LDX &H7002;STD 0,X;INX;INX;STX &H7002;PULX

8670 ASM X,XGDX;RORB;BCS O:REM BRANCH IF AUXILLIARY (ODD X)
8680 ASM X,ROLB;XGDX;OIM &H40,&H15;OIM &H01,&H17:REM PREPARE FOR AUX
8690 ASM X,LDX #&H7301;LDD &H7000;STD &H7002;JMP C:REM DO AUX

' EXIT AFTER DOING ACM AND AUX
8700 O=X:ASM X,ROLB;XGDX
8705 ASM X,CLR &H11:REM DISABLE SCI
8710 ASM X,LDAA #5;STAA &H10:REM BAUD RATE FROM TIMER1 (#5 GIVES 9600)
8720 ASM X,OIM &H42,&H1B:REM ENABLE TIMER2 INTS AND E/128
```

```
'8730 ASM X,LDAA #95;STAA &H1C:REM TIMER COUNT TO GET 100 Hz
8740 ASM X,OIM &H02,&H11:REM 02  RESTART UART NO RECEIVE
' reconf uart, ENABLE IT,  power off (PHYSICALLY ENABLE IT)
8750 ASM X,AIM &H14,&H15:REM PUT MULTIPLEXORS ON PARK
8760 ASM X,AIM &HFA,&H17:REM P60,P62=0,AUX & POWER OFF
8775 ASM X,LDD &H7002
8780 ASM X,RTS


9100 B=X:REM SUBTRACT AND TRANSFER SUBROUTINE
' POINTERS:  m(7000) = DATA ARRAY PERMANENT STORAGE. PASSED IN AB
'            m(7002) = FLAGGED AXIS PAIRS STORAGE
'                      6F38 IS @(50) IN RAM  MUST BE SAME AS A/D ROUTINE
'            m( X  ) = AUX LIST FOR END OF LIST CHECKING.  PASSED IN X

9110 ASM X, STD &H7000
9115 ASM X, LDD #&H6F38; STD &H7002
9120 K=X: ASM X,PSHX; LDX &H7002; LDD 0,X; INX;INX:REM GET WORD
9140 ASM X,BEQ F:REM TEST FLAG ON NORMAL MEAS

9150 ASM X,SUBD 0,X:REM DOUBLE SUBTRACT
9152 ASM X,RORA;RORB:REM SHIFT RIGHT WITH CARRY = DIVIDE BY TWO

9160 ASM X,TST 0,X:REM TEST FLAG ON REVERSED MEAS
9170 ASM X,BEQ F

9200 D=X  : ASM X, INX;INX;STX &H7002;PULX:REM MOVE ON TO NEXT PAIR

9203 REM ' SAVE RESULT TO DATAARRAY  AT m(7000)
9205 ASM X,PSHX;LDX &H7000;STD 0,X;INX;INX;STX &H7000;PULX

' CHECK FOR END OF LIST.  AUX LIST= FOUR BYTES / AXIS
9240 ASM X,INX;INX;INX;INX
9250 ASM X,LDAB #&HFF:REM CHECK FOR END OF LIST
9255 ASM X,EORB &H00,X
9260 ASM X,BNE K:REM LOOP IF NOT DONE
9265 ASM X,LDD &H7000:REM LOAD DATAFILE POINTER FOR RETURN

9270 ASM X,RTS:REM EXIT

9275 F=X
'REM FLAG MISSED
9280 ASM X,LDD #&H8000
9290 ASM X,BRA D

' FIND COMPASS N+1 TRANSMIT GAP
9500 T=X:ASM X,LDAA &H03:REM PORT 2
9510 D=X:ASM X,CMPA &H03;BEQ D:REM FIND EDGE
9520 ASM X,LDAA &H03;LDX #&H007B:REM 1ms GAP
9530 P=X:ASM X,CMPA &H03;BNE D;DEX;BNE P:REM FIND 1ms GAP
9540 ASM X,RTS

9900 RETURN

9999 END
```

```
10 PRINT " VORT2 BASS, 26-JUNE-94 ROM VERSION from IMCSBASS 17-March-94"
11 PRINT " 514 RECORDS EVERY 2 MIN FOR 16 BLOCKS (32 MIN) EVERY 2 HR,"
12 PRINT " EVERY 3 DAYS, EVERY OTHER MONTH."
'20 Q=&H4000:REM DATAFILE STARTS AT 4000 IN RAM
20 Q=&H0143:REM DATAFILE STARTS AT 0143 IN ROM

'F=PWR FLAG, 0 FOR NOTHING ON, 1 FOR P&S, 2 FOR P,S,&O.
'D=DO CALIBRATE A/D AND READ COMPASS LATCH
'ACMs,OBSs,PRES,& SEABIRDs MEASURED FOR 15 MINUTES EVERY HOUR AT 4 Hz.
'TO LATCH PWR, FIRST PWR ON AND WASTE TIME WITH:
'PSET 16:CALL &HFFD9,0:CALL &HFFD9,0, THEN:
'COMPASS ON      PSET 15
'PRESS ON        PSET 7,14:PCLR 7,16,14
'SEABIRDS ON     PSET 6,14:PCLR 6,16,14
'OBSs ON         PSET 2,14,:PCLR 2,16,14.
'TO PREVENT TURNING THESE OFF WHEN SELECTING AUX WITH D\6,
'HOLD D\4 HI WITH PSET 16:CALL &HFFD9,0:CALL &HFFD9,0
'PSET 1,14:PCLR 1,16 WHERE CALL &HFFD9 IS A 2.2MS TEST OF UART BUFFER.
100 ASM &H9E,DB &H00:REM CONTROL DISABLE
102 SLEEP 0:PCLR 0,1,2,3,4,5,6,7,8,14,15:PSET 14,16:REM ESTABLISH DDRs
108 SLEEP 240:PCLR 14,16:REM LET A/D CALIBRATE FINISH IF STARTED.
110 PRINT "   TYPE   WAKE   TO GET CONTROL (5sec)"
112 X = 0: STORE X,#4,&H01010101 : X = 0: ITEXT X,500
114 X = 0: IF GET(X,#4)<>&H57414B45 GOTO 300

130 RTIME
131 PRINT #02,"OLD TIME=",?(3),"/",?(4),"/",?(5)," ",?(2),":",?(1),":",?(0)
132 INPUT " SET TIME? 1=YES",J
133 IF J<> 1 GOTO 160
134 INPUT " THE YEAR 0-99    ",?(5)
136 INPUT " THE MONTH 1-12   ",?(4)
138 INPUT " THE DAY 1-31   ",?(3)
140 INPUT " THE HOUR 0-23    ",?(2)
142 INPUT " THE MINUTE 0-59 ",?(1)
144 INPUT " THE SECOND 0-59 ",?(0)
150 STIME
160 RTIME
162 PRINT #02,"NEW TIME=",?(3),"/",?(4),"/",?(5)," ",?(2),":",?(1),":",?(0)
164 INPUT " STOP? 1=YES",J
166 IF J<> 1 GOTO 300
230 STOP

300 ONERR 100
1000 DIM(1024,8192)
' ASSEMBLY ROUTINES
1010 X=0:A=0:B=0:C=0:D=0:E=514:F=0:G=0:H=0:I=0:J=0:K=0:L=0:M=0:N=0:O=0
1015 PRINT "    A    B    C    D    E    F    G    H";
1016 PRINT "    I    J    K    L    M    N    O    P.    T    X"
1020 GOSUB 8000:REM FIRST PASS
1025 PRINT #5H,A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,T,X
1030 GOSUB 8000:REM SECOND PASS
1031 PRINT #5H,A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,T,X
1050 N=0:REM INITIALIZE COUNTER
1080 GOTO 2000

'SUBROUTINE TO TURN ON PRESSURE AND SEABIRDS
1200 PSET 16:CALL &HFFD9,0:CALL &HFFD9,0:REM LET PWR SETTLE
1210 PSET 6,7,14:PCLR 6,7,16,14:F=1:RETURN
```

```
'SUBROUTINE TO TURN ON OBSs, PRESSURE, AND SEABIRDS
1230 PSET 16:CALL &HFFD9,0:CALL &HFFD9,0:REM LET PWR SETTLE
1240 PSET 2,6,7,14:PCLR 2,6,7,16,14:F=2:RETURN

'SUBROUTINE TO TURN OFF EVERYTHING
1250 PCLR 2,6,7,14:PSET 14:PCLR 14:F=0:RETURN

'SUBROUTINE TO ENABLE AUXILLIARIES ON MULTIPLEXOR
1260 PSET 16:REM LET PWR SETTLE
1270 PSET 1,14:PCLR 1,16:RETURN:REM DISABLES POWER LATCH

' CALIBRATE A/D
2000 SLEEP 0:PSET 15:SLEEP 600:PSET 16:REM PWR ON,CLEAR LATCHED PWR
2050 CALL T,0:REM FIND COMPASS GAP
2060 Z=COUNT(40)-1:SLEEP 50:REM Z=aZimuth,COMPASS
'2065 PRINT "GOT COMPASS GAP"
2070 PCLR 15:SLEEP 240:REM A/D CALIBRATE;2,882,040 CYCLES AT 1.2288 MHz
2080 PCLR 14,16:REM POWER OFF AND PREPARE TO LATCH PWR
2082 D=0:REM CLEAR LATCH TO DO A/D AND Zn
2090 GOSUB 1200:REM TURN ON SEABIRD,PRESS
2100 SLEEP 0:P=SDI(24):REM SAMPLE AND READ PRESSURE

3000 RTIME:IF E>0 GOTO 3040
3001 IF ?(4)%2=0 GOTO 3030:REM RUN IN JA,MAR,MAY,JUL,S (JUL,S,N,JA,MAR)
3006 IF ?(3)%3<>1 GOTO 3030:REM RUN EVERY THIRD DAY
3008 IF ?(2)%2>0 GOTO 3030:REM ONLY RUN EVERY OTHER HOUR
3010 IF ?(1)>32 D=1:GOTO 3030:REM OFF MINUTE 33 TO END OF HOUR
3012 IF ?(0)>0 GOTO 3030:REM CONTINUE
3014 IF ?(1)%2=0 E=514
3015 IF D=1 GOTO 2000:REM COMPASS AND A/D ONCE/HR
3020 GOTO 2090:REM POWER ON PRESSURE AND START CYCLE

3030 GOSUB 1250:SLEEP 100:GOTO 3000:REM EVERYTHING OFF

3040 E=E-1:X=0:REM INITIALIZE DATAFILE
3050 STORE X,#2,&HAB52:REM KEY WORD AB V2 FOR VORT2 OR AB52
3051 STORE X,#1,?(4):REM MONTH
3052 STORE X,#1,?(3):REM DAYS
3055 STORE X,#1,?(2):REM HOURS
3060 STORE X,#1,?(1):REM MINUTES
3070 STORE X,#1,?(0):REM SECONDS
3080 STORE X,#1,N:REM COUNTER

3100 C=&H73000000+Q+X:REM MUX ADDRESS AND DATAFILE POINTER
3110 CALL A,C,X:REM CALL A/D ROUTINE,ACM THEN AUX,POWER OFF AT END
3111 X=X%&H10000-Q:REM GET BACK THE DATAFILE POINTER
' SUBTRACT ROUTINE (CALL B) EXPECTS FLAGGED DATA AT 6F38.
3119 C=&H73000000+Q+X:REM DATAFILE LOCATION
3120 CALL B,C,X:REM CALL SUBTRACT AND TRANSFER
3121 X=X%&H10000-Q:REM GET BACK THE BASIC DATAFILE POINTER
3125 STORE X,#2,Z:REM COMPASS
3130 STORE X,#4,P:REM PRESSURE

'3170 X=0:REM REINITIALIZE DATAFILE
'3180 PRINT #4H,GET(X,#2):REM KEYWORD
'3185 PRINT #02,GET(X,#1)," ",GET(X,#1)," ";:REM MONTH DAY
'3190 PRINT #02,GET(X,#1),":",GET(X,#1),":",GET(X,#1)," ";:REM HR:MM:SS
'3200 PRINT #02H, GET(X,#1):REM COUNT
'3205 FOR L=1 TO 6:REM T(0:3),PITCH,ROLL
```

```
'3206 PRINT #04H,GET(X,#2)," ";
'3207 NEXT L:PRINT
'3210 FOR M=1 TO 6:REM 2 VORT SENSORS
'3220 FOR L=1 TO 4:REM 4 WORDS/LINE
'3230 PRINT #04H,GET(X,#2)," ";:REM EACH SENSOR ON A LINE
'3240 NEXT L:PRINT:REM FOUR AXES/LINE
'3250 NEXT M
'3262 PRINT #3,GET(X,#2)," ";:REM COMPASS
'3264 PRINT #08H,GET(X,#4)," ";:REM PRESSURE
'3275 PRINT:REM SPACE
3280 PRINT {0,73}::REM TAKES 77 ms

3290 N=N+1
3295 SLEEP 14:P=SDI(24):REM SAMPLE COUNTERS AND READ PRESSURE
3300 GOTO 3000

8000 X=&H7300:REM MULTIPLEXOR LIST
8010 ASM X,DW &H0000;DW &H0101;DW &H0202;DW &H0303:REM ACM 1/T1-T4
8015 ASM X,DW &H088A;DW &H098B;DW &H0AFF;DW &H0BFF:REM ACM 1/PITCH, ROLL
8020 ASM X,DW &H20FF;DW &H21FF;DW &H22FF;DW &H23FF:REM ACM 2
8025 ASM X,DW &H28FF;DW &H29FF;DW &H2AFF;DW &H2BFF
8030 ASM X,DW &H4000;DW &H4100;DW &H4200;DW &H4300:REM ACM 3
8035 ASM X,DW &H4800;DW &H4900;DW &H4A00;DW &H4B00
8040 ASM X,DW &H6000;DW &H6100;DW &H6200;DW &H6300:REM ACM 4
8045 ASM X,DW &H6800;DW &H6900;DW &H6A00;DW &H6B00
8050 ASM X,DW &H8000;DW &H8100;DW &H8200;DW &H8300:REM ACM 5
8060 ASM X,DW &H8800;DW &H8900;DW &H8A00;DW &H8B00
8070 ASM X,DW &HA000;DW &HA100;DW &HA200;DW &HA300:REM ACM 6
8080 ASM X,DW &HA800;DW &HA900;DW &HAA00;DW &HAB00
8090 ASM X,DW &HFF00;DW &HFF00:REM END OF LIST

8100 X=&H3800:A=X:REM A/D ROUTINE
'   X IS POINTER TO AUX ARRAY, AB IS POINTER TO FLAGGED CONVERSION STORAGE
8105 ASM X,STD &H7000:REM POINTER TO DATAFILE
8107 ASM X,LDD #&H6F38;STD &H7002:REM POINTER TO FLAGGED CONVERSIONS
8110 ASM X,OIM &H04,&H17;PSHX;LDX #&H0600:REM POWER ON, WAIT 5ms
8120 D=X:ASM X,DEX;BNE D;PULX:REM WAIT (4*.81us/LOOP)
8150 ASM X,CLR &H11;LDAA #&H10;STAA &H10:REM DISABLE UART, SET SCI FOR
8160 ASM X,OIM 8,&H11:REM INTERNALLY CLOCKED 8 BIT DATA RECEIVE ENABLE
'DUMMY PULSE SENT OUT TO INITIALIZE
8170 ASM X,OIM &HAB,&H15;AIM &HBF,&H15:REM DUMMY AUX LOCATION
8171 ASM X,OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03
'START TIMING P27=1
8172 ASM X,AIM &H7F,&H03:REM REMOVE PULSE P27=0
8174 D=X:ASM X,LDAB #&H02;BITB &H03;BEQ D:REM CHECK P21 FOR DONE
8179 C=X:ASM X,AIM &H14,&H15:REM ENTRY POINT FOR AUX,MASK PORT 5
8180 ASM X,LDAA &H00,X;INX;INX:REM TOP OF MUX LIST,MOVE TO NEXT
8185 ASM X,ORAA &H15;STAA &H15:REM PUT MUX ON PORT 5
8200 ASM X,OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03
'START TIMING P27=1
8210 ASM X,AIM &H7F,&H03:REM REMOVE PULSE P27=0

8220 D=X:ASM X,LDAB #&H02;BITB &H03;BEQ D:REM CHECK P21 FOR DONE

8240 ASM X,LDAB &H15;ANDB #&H14:REM MASKED PORT 5 IN B
8250 ASM X,LDAA &H00,X;INX;INX:REM GET NEXT MUX WORD,ADVANCE
8255 ASM X,ABA;STAA &H15:REM ADD MUX WORD TO MASKED PORT 5, STORE P5
8280 ASM X,OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03;OIM &H80,&H03
'
```

```
START TIMING P27=1
8290 ASM X,AIM &H7F,&H03:REM REMOVE PULSE P27=0

8400 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8402 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8403 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8404 ASM X,CPX 0;CPX 0;CPX 0;CPX 0

8410 ASM X,LDAA &H11;LDAB &H12:REM TRIGGER A READ
8420 ASM X,CPX 0;CPX 0;CPX 0;CPX 0:REM 16ECLOCK
8430 ASM X,LDAA &H11;LDAB &H12:REM HIGH BYTE IN B AND READ NEXT
8440 ASM X,CPX 0;CPX 0;CPX 0;CPX 0:REM 16ECLOCK
8450 ASM X,LDAA &H12:REM LOW BYTE IN A  DON'T READ NEXT
'REM FLIP THE BITS OF THE WORDS
8451 ASM X,ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA;RORB
8452 ASM X,ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA

8480 ASM X,TIM &H20,&H03;BEQ G:REM TEST BOTH RECEIVED
8488 ASM X,XGDX;RORB;BCS M:REM BRANCH IF AUXILLIARY (ODD X)
8490 ASM X,ROLB;XGDX;CLRA;CLRB;BRA G:REM IF NOT RECEIVED,CLEAR
8495 M=X:ASM X,ROLB;XGDX
' PUT AWAY TO @(50) ARRAY  THE FLAGGED VALUES
8510 G=X:ASM X,PSHX;LDX &H7002;STD 0,X;INX;INX;STX &H7002;PULX

8520 ASM X,LDAB #&HFF;EORB &H00,X;BEQ L;JMP D:REM CHECK FOR END OF LIST

8550 L=X:ASM X,LDAB #&H02:REM TEST FOR HOLD PULSE
8555 D=X:ASM X,BITB &H03;BEQ D:REM CHECK P21

8572 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8573 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8574 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8575 ASM X,CPX 0;CPX 0;CPX 0;CPX 0
8580 ASM X,LDAA &H11; LDAB &H12:REM TRIGGER A READ
8590 ASM X,CPX 0;CPX 0;CPX 0;CPX 0:REM 16ECLOCK
8600 ASM X,LDAA &H11;LDAB &H12:REM HIGH BYTE IN B AND READ NEXT
8610 ASM X,CPX 0;CPX 0;CPX 0;CPX 0:REM 16ECLOCK
8620 ASM X,LDAA &H12:REM LOW BYTE IN A DON'T READ NEXT
'REM FLIP THE BITS OF THE WORDS PUTS HIGH BYTE IN A, LOW BYTE IN B
8621 ASM X,ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA;RORB
8622 ASM X,ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA;RORB;ROLA

8630 ASM X,TIM &H20,&H03;BEQ H:REM TEST BOTH RECEIVED
8640 ASM X,XGDX;RORB;BCS N:REM BRANCH IF AUXILLIARY (ODD X)
8645 ASM X,ROLB;XGDX;CLRA;CLRB;BRA H:REM IF NOT RECEIVED,CLEAR
8650 N=X:ASM X,ROLB;XGDX

' FINISH PUT AWAY TO @(50) ARRAY  THE FLAGGED VALUES
8660 H=X:ASM X,PSHX;LDX &H7002;STD 0,X;INX;INX;STX &H7002;PULX

8670 ASM X,XGDX;RORB;BCS O:REM BRANCH IF AUXILLIARY (ODD X)
8680 ASM X,ROLB;XGDX;OIM &H40,&H15;OIM &H01,&H17:REM PREPARE FOR AUX
8690 ASM X,LDX #&H7301;LDD &H7000;STD &H7002;JMP C:REM DO AUX

' EXIT AFTER DOING ACM AND AUX
8700 O=X:ASM X,ROLB;XGDX
8705 ASM X,CLR &H11:REM DISABLE SCI
8710 ASM X,LDAA #5;STAA &H10:REM BAUD RATE FROM TIMER1 (#5 GIVES 9600)
8720 ASM X,OIM &H42,&H1B:REM ENABLE TIMER2 INTS AND E/128
```

```
'8730 ASM X,LDAA #95;STAA &H1C:REM TIMER COUNT TO GET 100 Hz
8740 ASM X,OIM &H02,&H11:REM 02  RESTART UART NO RECEIVE
' reconf uart, ENABLE IT,  power off (PHYSICALLY ENABLE IT)
8750 ASM X,AIM &H14,&H15:REM PUT MULTIPLEXORS ON PARK
8760 ASM X,AIM &HFA,&H17:REM P60,P62=0,AUX & POWER OFF
8775 ASM X,LDD &H7002
8780 ASM X,RTS


9100 B=X:REM SUBTRACT AND TRANSFER SUBROUTINE
' POINTERS:  m(7000) = DATA ARRAY PERMANENT STORAGE. PASSED IN AB
'            m(7002) = FLAGGED AXIS PAIRS STORAGE
'                 6F38 IS @(50) IN RAM  MUST BE SAME AS A/D ROUTINE
'            m( X  ) = AUX LIST FOR END OF LIST CHECKING.  PASSED IN X

9110 ASM X, STD &H7000
9115 ASM X, LDD #&H6F38; STD &H7002
9120 K=X: ASM X,PSHX; LDX &H7002; LDD 0,X; INX;INX:REM GET WORD
9140 ASM X,BEQ F:REM TEST FLAG ON NORMAL MEAS

9150 ASM X,SUBD 0,X:REM DOUBLE SUBTRACT
9152 ASM X,RORA;RORB:REM SHIFT RIGHT WITH CARRY = DIVIDE BY TWO

9160 ASM X,TST 0,X:REM TEST FLAG ON REVERSED MEAS
9170 ASM X,BEQ F

9200 D=X  : ASM X, INX;INX;STX &H7002;PULX:REM MOVE ON TO NEXT PAIR

9203 REM ' SAVE RESULT TO DATAARRAY  AT m(7000)
9205 ASM X,PSHX;LDX &H7000;STD 0,X;INX;INX;STX &H7000;PULX

' CHECK FOR END OF LIST.  AUX LIST= FOUR BYTES / AXIS
9240 ASM X,INX;INX;INX;INX
9250 ASM X,LDAB #&HFF:REM CHECK FOR END OF LIST
9255 ASM X,EORB &H00,X
9260 ASM X,BNE K:REM LOOP IF NOT DONE
9265 ASM X,LDD &H7000:REM LOAD DATAFILE POINTER FOR RETURN

9270 ASM X,RTS:REM EXIT

9275 F=X
'REM FLAG MISSED
9280 ASM X,LDD #&H8000
9290 ASM X,BRA D

' FIND COMPASS N+1 TRANSMIT GAP
9500 T=X:ASM X,LDAA &H03:REM PORT 2
9510 D=X:ASM X,CMPA &H03;BEQ D:REM FIND EDGE
9520 ASM X,LDAA &H03;LDX #&H007B:REM 1ms GAP
9530 P=X:ASM X,CMPA &H03;BNE D;DEX;BNE P:REM FIND 1ms GAP
9540 ASM X,RTS

9900 RETURN

9999 END
```

DOCUMENT LIBRARY

Distribution List for Technical Report Exchange – July 1998

University of California, San Diego
SIO Library 0175C
9500 Gilman Drive
La Jolla, CA 92093-0175

Hancock Library of Biology & Oceanography
Alan Hancock Laboratory
University of Southern California
University Park
Los Angeles, CA 90089-0371

Gifts & Exchanges
Library
Bedford Institute of Oceanography
P.O. Box 1006
Dartmouth, NS, B2Y 4A2, CANADA

NOAA/EDIS Miami Library Center
4301 Rickenbacker Causeway
Miami, FL 33149

Research Library
U.S. Army Corps of Engineers
Waterways Experiment Station
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

Marine Resources Information Center
Building E38-320
MIT
Cambridge, MA 02139

Library
Lamont-Doherty Geological Observatory
Columbia University
Palisades, NY z10964

Library
Serials Department
Oregon State University
Corvallis, OR 97331

Pell Marine Science Library
University of Rhode Island
Narragansett Bay Campus
Narragansett, RI 02882

Working Collection
Texas A&M University
Dept. of Oceanography
College Station, TX 77843

Fisheries-Oceanography Library
151 Oceanography Teaching Bldg.
University of Washington
Seattle, WA 98195

Library
R.S.M.A.S.
University of Miami
4600 Rickenbacker Causeway
Miami, FL 33149

Maury Oceanographic Library
Naval Oceanographic Office
Building 1003 South
1002 Balch Blvd.
Stennis Space Center, MS, 39522-5001

Library
Institute of Ocean Sciences
P.O. Box 6000
Sidney, B.C. V8L 4B2
CANADA

National Oceanographic Library
Southampton Oceanography Centre
European Way
Southampton SO14 3ZH
UK

The Librarian
CSIRO Marine Laboratories
G.P.O. Box 1538
Hobart, Tasmania
AUSTRALIA 7001

Library
Proudman Oceanographic Laboratory
Bidston Observatory
Birkenhead
Merseyside L43 7 RA
UNITED KINGDOM

IFREMER
Centre de Brest
Service Documentation - Publications
BP 70 29280 PLOUZANE
FRANCE

50272-101

REPORT DOCUMENTATION PAGE	1. REPORT NO. WHOI-98-13	2.	3. Recipient's Accession No.

4. Title and Subtitle Vorticity Measurements within the Bottom Boundary Layer in the Strait of Juan De Fuca	5. Report Date July 1998
	6.
7. Author(s) J.J. Fredericks, John H. Trowbridge, A.J. Williams, 3rd	8. Performing Organization Rept. No. WHOI-98-13
9. Performing Organization Name and Address Woods Hole Oceanographic Institution Woods Hole, Massachusetts 02543	10. Project/Task/Work Unit No.
	11. Contract(C) or Grant(G) No. (C) N00014-94-I-0436 (G)
12. Sponsoring Organization Name and Address Office of Naval Research	13. Type of Report & Period Covered Technical Report
	14.

15. Supplementary Notes

This report should be cited as: Woods Hole Oceanog. Inst. Tech. Rept., WHOI-98-13.

16. Abstract (Limit: 200 words)

Electromagnetic fluctuations and turbulent vorticity fluctuations were measured over a nine month period in the strong tidal flows of the Strait of Juan De Fuca off the coast of the Olympic Peninsula of Washington. A collaborative experiment was designed to test the hypothesis that electromagnetic fluctuations at the sea floor are forced by turbulent vorticity fluctuations in the bottom boundary layer. This report describes the measurement of turbulent vorticity fluctuations and the associated analysis which focuses on testing existing theoretical predictions for the inertial subrange and on characterizing spectra at frequencies below the inertial subrange.

17. Document Analysis a. Descriptors

vorticity
electromagnetic fluctuations
bottom boundary layer
turbulent fluctuation

b. Identifiers/Open-Ended Terms

c. COSATI Field/Group

18. Availability Statement	19. Security Class (This Report)	21. No. of Pages
Approved for public release; distribution	UNCLASSIFIED	75
	20. Security Class (This Page)	22. Price

(See ANSI-Z39.18) *See Instructions on Reverse* OPTIONAL FORM 272 (4-77)
(Formerly NTIS-35)
Department of Commerce